TOWARDS A FULLY SCALABLE BALANCED PARAREAL METHOD: APPLICATION TO NEUTRONICS

Yvon Maday, Olga Mula, Mohamed-Kamel Riahi

TOWARDS A FULLY SCALABLE BALANCED PAREAL METHOD: APPLICATION TO NEUTRONICS *

Y. MADAY*†‡, O. MULA§, AND M. K. RIAHI¶

Abstract. In the search of new approaches for the efficient exploitation of large scale computational platforms, the parallelization in the time direction for time dependent problems is a very promising approach. Among the existing methods in this frame, the parallel in time method, since its introduction in [10], has been developed in many ways that, altogether, allow to identify its pros and cons. Among the cons, the current approaches present in practice some efficiency limitations as regards correct scalings that "spoil" the huge potential of the idea. This article is a contribution towards overcoming this major obstruction by exploiting the idea that the numerical schemes to parallelize time could be coupled to other iterative numerical algorithms that are needed to solve the PDE. We present a parareal scheme in which these alternative iterations are truncated (i.e. not converged) during each parareal iteration but in which convergence is nevertheless achieved across the parareal iterations. In order to limit the use of too much memory necessitated by the recovery of these alternative iterations over the parareal iterations, we propose also a compression procedure via proper orthogonal decomposition. After a mathematical analysis of the convergence properties of this new approach, we present some numerical results dealing with the application of the scheme to accelerate the time-dependent neutron diffusion equation in a reactor core. The numerical results show a significant improvement of the performances with respect to the plain parareal algorithm, which is an important step towards making the parallelization in the time direction be a fully competitive option for the exploitation of massively parallel computers.

Key words. parareal in time algorithm, degraded fine solver, fixed-point iterations, reduced basis, neutronics, neutron diffusion equation

AMS subject classifications. AMS subject classifications. 65M12, 65N55, 65Y05

Introduction. In the framework of the approximation of physical or industrial processes, the number of computing cores that can be dedicated to the simulation of a particular phenomenon is nowadays very large. Although this already allows to simulate processes of increasing complexity with an increasing accuracy within a reasonable clock time, the efficient exploitation of these large scale systems requires the development of numerical algorithms that present good scalability properties on the current architecture and should maintain these properties on the next generation platforms. The idea of task decomposition plays a major role when it comes to address this issue. In particular, spatial domain decomposition is a very active research area with impressive success stories in by now a large variety of application fields (see [15] for an overview). However, it is well-known that its optimal scalability properties are progressively degraded (and end up by stagnating) as the number of cores/subdomains increases, hence the need to search for additional parallelism directions. One of the reasons for this degradation is that, as the number of cores/subdomains increases, the

*This work was supported in part by the joint research program MANON between CEA-Saclay and University Pierre et Marie Curie-Paris 6. The authors would like to thank Frédéric Hecht for useful advices with the implementations with Freefem++. Thank you also to Philippe Parnaudeau for helping in the access to the computing facilities at Laboratoire Jacques-Louis Lions/UPMC.

*Sorbonne Universités, UPMC Univ Paris 06, CNRS, UMR 7598, Laboratoire Jacques-Louis Lions, 4, place Jussieu 75005, Paris, France. (email: maday@ann.jussieu.fr)

†Institut Universitaire de France

‡Division of Applied Mathematics, Brown University, Providence RI, USA.

§Aachen Institute for Advanced Study in Computational Engineering Science (AICES) Graduate School, Schinkelstrasse 2, 52056 Aachen, Germany (email: mula@aices.rwth-aachen.de)

¶Department of Mathematical Science, New Jersey Institute of Technology, NJ-07102 New Jersey, USA (email: riahi@njit.edu)

size of each subdomain tends to diminish and thus the amount of work to perform on each node become small with respect to the inter-communication time. In this respect, the parallelization of the time domain carries a lot of potential since most simulations which are expected to deliver economic, societal and scientific impact from large scale systems contain time-stepping in some form. While several approaches have been proposed over the years to decompose the time direction (see [5] for an overview), they all suffer from severe efficiency limitations which "spoil" the huge potential of the idea. In this paper, we would like to contribute to overcome this major obstruction by exploiting the idea that the numerical schemes to parallelize time could be coupled to other schemes that are needed for the complete resolution of the problem under consideration.

We will focus on improving the performances of the so-called parareal in time algorithm (first introduced in [10]), that has been developed over the years in many ways and has allow to build a community in this effort of parallelizing in the time direction. The parareal in time algorithm is a domain decomposition technique that is based on a prediction-correction strategy in which the prediction is performed by a coarse (and thus computationally cheap) solver $\mathcal{G}$ and the correction is done by using an accurate (but computationally expensive) solver $\mathcal{F}$. If the problem is formulated over an interval $[0, T]$ and $(T_N)_{N=0,\dots,\underline{N}}$ is a set of increasing times in this interval, then the parareal in time method aims at building a sequence of approximations $(U_N^k)_k$ of the exact solution $u(T_N)$ at each time T_N such that, as k goes to infinity, U_N^k converges to $u(T_N)$. As it will be recalled in section 1, the sequence $(U_N^k)_k$ is defined through a recursion formula involving predictions through $\mathcal{G}$ and corrections through $\mathcal{F}$. The method decomposes the time domain in the sense that it allows to divide the propagation of the fine solver over $[0, T]$ into propagations on independent sub-intervals $[T_N, T_{N+1}]$ that can be run concurrently on several processors.

As has already been brought up many times in the literature (see, e.g., [2, 1]), it is well-known that, if k parareal iterations are required to achieve convergence, then the optimal speedup using $\underline{N}$ processors is roughly of order $\underline{N}/k$. Since the method usually converges in about three or four parareal iterations, it implies that the efficiency will be around $1/3$ or $1/4$, which may already be interesting for big systems of ODE's but nevertheless may not be sufficient to justify the investment for a larger parallel platform. The challenge of overcoming this efficiency limitation has already been addressed in several previous contributions such as in [12, 11, 13]. They are all based in diminishing the cost of the fine solver, and take benefit of the iterative process in order to progressively improve the realization of the fine solver across the parareal iterations. For example, in [12] (see also [6]), it has been proposed to use a spatial domain decomposition algorithm to realize the fine solver in which the number of (spatial domain decomposition) iterations is limited on each (parareal) iteration. Then, in the following parareal iteration, the spatial iterations are resumed by using the previous state as an initial guess in the spatial domain decomposition iterations. This idea can actually be extended to any type of other iterative procedures like, e.g., optimal control [12] or high order time stepping [13].

In this paper, we propose to adapt this strategy at the algebraic level, i.e., we place ourselves in the case where the propagation of the fine solver requires the inversion of a system through an iterative procedure and we truncate these "internal" iterations on each parareal iteration. Although the results that will be presented involve a specific form of this iterative process (Jacobi or Gauss-Seidel type methods), we point out that our procedure is general and could be applied to any type of linear fixed-point

scheme. With this particular form of iterative schemes, our main result is that, under several hypothesis, convergence is probably achieved across the parareal iterations in spite of the "local" degradation of the fine solver. This part of the work is presented in sections 2 and 3.

The rest of the paper is then devoted to practical issues that arise in the implementation and to the application of the method to a numerical example. In section 4, we see that the practical realization of this new version of the parareal algorithm presents an important obstruction. Indeed, for a given parareal iteration, the initialization of the internal iterations at every time step requires the knowledge of the fine states of the previous parareal iteration. It is then necessary to store all the previous fine states, which might be an unfeasible requirement in large problems. We explain how the use of complexity reduction techniques could be an efficient strategy to address this memory storage issue.

Finally, in section 5, we apply our method on the time-depending neutron diffusion model, which is a non-trivial computational model widely used in nuclear industry as it approximate the very-complex Boltzmann's neutrons transport model. We then study the efficiency of our method. Last but not least, we illustrate the feasibility of "compressing" the information contained by the fine solutions in order to overcome memory storage limitations. For the reasons that are explained in section 5, we consider that our numerical application is a preliminary study towards the application of the method in an involved non-academic framework, namely the resolution of the aforementioned neutron transport equation in industrial codes. In this respect, the present work is a sequel of [9] in which the plain parareal method was implemented in a production code and it will serve as a basis for implementing our parareal method in this more involved framework in a future work.

1. The plain parareal algorithm. Let us introduce some notation and recall more in detail the mechanisms of the plain parareal in time algorithm. Let us consider an evolution problem over an interval $[0, T]$ that reads: Find a time dependent vector $u(t) \in \mathbb{R}^{\mathcal{N}}$ solution to the following problem

$$\begin{cases} \dfrac{\mathrm{d}u(t)}{\mathrm{dt}} + \mathcal{A}(t)u(t) = f(t) \\ u(0) = u_0 \end{cases} \tag{1.1}$$

where $f \in \mathbb{R}^{\mathcal{N}}$ is given. This problem is either a differential system of dimension $\mathcal{N}$ or may come from the spatial discretization of a parabolic linear Partial Differential Equation (PDE), then, $\mathcal{N}$ denotes the number of degrees of freedom of the space approximation, and $\mathcal{A} \in \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ is the inverse of the mass matrix multiplied by the stiffness matrix. Note that f and $\mathcal{A}$ may depend on time, hence the notation $f(t)$, $\mathcal{A}(t)$.

Let $s > 0$ denote a time step and let $\tau = ns$ be a time range, $n \in \mathbb{N}^*$. For a fixed τ, any discretization in time of (1.1) based on the time step s gives rise to a discrete propagator $\mathcal{P}_\tau^t$, that can be implicit or explicit, of Euler, Runge Kutta, Adams-Bashforth-Moulton, ... type. Starting from any given "initial" condition at time t, $\mathcal{P}_\tau^t$ provides an approximation of the solution of (1.1) at time $t+\tau$. If the time step s is small enough, the approximation will be accurate, we then denote by $\mathcal{F}_\tau^t$ such a fine discrete propagator. The accuracy of $\mathcal{P}_\tau^t$ can be degraded by using larger time steps $s' > s$ or a more simple discretization scheme. It can also be degraded by "simplifying" the approximation of the phenomenon under consideration (e.g. by

considering a model with reduced physics). Let us denote by $\mathcal{G}_\tau^t$ such a coarse discrete propagator.

Let be given a decomposition of the solution time interval $[0,T]$ into $\underline{\mathrm{N}}$ time intervals $[T_N, T_{N+1}]$, $N = 0, \ldots, \underline{\mathrm{N}} - 1$ of — say — uniform size $\Delta T = T_{N+1} - T_N$. The parareal in time algorithm — in its plain version — proposes a series $(U_k^N)_k$ of approximations of the solution $u(T_N)$ of (1.1) at time T_N. The algorithm reads

$$\begin{cases} U_{k+1}^{N+1} = \mathcal{G}_{\Delta T}^{T_N}(U_{k+1}^N) + \mathcal{F}_{\Delta T}^{T_N}(U_k^N) - \mathcal{G}_{\Delta T}^{T_N}(U_k^N), & k \geq 0, \quad 0 \leq N \leq \underline{N} - 1 \\ U_0^{N+1} = \mathcal{G}_{\Delta T}^{T_N}(U_0^N), & 0 \leq N \leq \underline{N} - 1 \\ U_0^0 = u_0. \end{cases} \tag{1.2}$$

As k goes to infinity, $(U_k^N)_k$ converges to $\mathcal{F}_{T_N}^0(u_0)$, which is the fine approximation of $u(T_N)$.

Since the algorithm converges to $\mathcal{F}_{T_N}^0(u_0)$ and not to the exact solution $u(T_N)$, the accuracy of the approximation U_k^N is at most the accuracy of the fine solver with respect to the true solution. Denoting by $\varepsilon_{\mathcal{F}}$ this accuracy, the parareal iterations should be stopped whenever

$$\max_{0 \leq N \leq \underline{N}} \|U_k^N - \mathcal{F}_{T_N}^0(u_0)\| \simeq \varepsilon_{\mathcal{F}}, \tag{1.3}$$

where $\|.\|$ denotes the euclidean norm in $\mathbb{R}^{\mathcal{N}}$.

Remark 1.1. *The method can be seen as a predictor-corrector strategy in which the prediction is given by $\mathcal{G}_{\Delta T}^{T_N}(U_{k+1}^N)$ and the correction is $\mathcal{F}_{\Delta T}^{T_N}(U_k^N) - \mathcal{G}_{\Delta T}^{T_N}(U_k^N)$. It can also be viewed as an extrapolation of $\mathcal{F}_{\Delta T}^{T_N}(U_k^N)$ by the term $\mathcal{G}_{\Delta T}^{T_N}(U_{k+1}^N) - \mathcal{G}_{\Delta T}^{T_N}(U_k^N)$ which allows to correct the fact that the fine propagation $\mathcal{F}_{\Delta T}^{T_N}$ should act on U_{k+1}^N and not on U_k^N.*

2. Definition of the Scalable Balanced Parareal Method : truncated internal iterations. For the sake of simplicity, we shall assume in what follows that, for any for any $N \in \{0, \ldots, \underline{\mathrm{N}} - 1\}$, $\mathcal{F}_{\Delta T}^{T_N}$ uses an Euler backward time discretization with time step δt and $\underline{\mathrm{n}}$ is the total number of time steps to propagate the solution between T_N and T_{N+1} : we therefore have the relation $\Delta T = \underline{\mathrm{n}}\delta t$. As a result, we have $T = \underline{\mathrm{N}}\Delta T = \underline{\mathrm{N}}\ \underline{\mathrm{n}}\delta t$ and the total number of fine propagations in the whole interval $[0,T]$ is $\underline{\underline{\mathrm{n}}} = \underline{\mathrm{N}}\ \underline{\mathrm{n}}$.

For $n \in \{0, \ldots, \underline{\underline{\mathrm{n}}}\}$, let u^n be the approximation of $u(t^n)$ with this fine propagator for $t^n = n\delta t$. The discretized version of problem (1.1) then reads as a loop over n, $n \in \{0, 1, \ldots, \underline{\underline{\mathrm{n}}} - 1\}$:

$$\begin{array}{l} \text{Given } u^n \in \mathbb{R}^{\mathcal{N}}, \text{ find } u^{n+1} \in \mathbb{R}^{\mathcal{N}} \text{ such that} \\ \begin{cases} Au^{n+1} = Bu^n + f^n, \\ u^0 = u_0, \end{cases} \end{array} \tag{2.1}$$

where $f^n = f(t^n)$, $A = \frac{\mathrm{Id}}{\delta t} + \mathcal{A}(t^{n+1})$ and $B = \frac{\mathrm{Id}}{\delta t}$, where Id denotes the identity matrix. In order not to overload the notations, A and B will be taken constant in this section. The resolution of problem (2.1) requires the inversion of A. Although the most suitable techniques to do this inversion depend on the problem under consideration, in general they involve iterative methods. In the present paper, we have written the theory in the case of Jacobi or Gauss-Seidel algorithms, but our procedure could be adapted to other (more involved) fixed-point schemes. In the Jacobi or Gauss-Seidel case,

A is splitted into D and $A-D$, where D may be $D = \mathrm{diag}[A] = \mathrm{diag}[\frac{\mathrm{Id}}{\delta t} + \mathcal{A}]$, $D = \frac{\mathrm{Id}}{\delta t} + \mathrm{diag}[\mathcal{A}]$ or even $D = \frac{\mathrm{Id}}{\delta t}$ in the Jacobi case.We are lead to solve

$$\forall n,\ 0 \leq n \leq \underline{\underline{\mathrm{n}}} - 1, \quad \forall j,\ 1 \leq j \leq J^*_{n+1},$$
$$u^{n+1,j} = (\mathrm{Id} - D^{-1}A)u^{n+1,j-1} + D^{-1}Bu^{n,J^*_n} + D^{-1}f^n$$

starting from

$$u^{n+1,0} = u^{n,J^*_n},$$

with a maximum of J^*_{n+1} iterations, sufficiently large to guarantee convergence. As a matter of self-consistency, $u^{0,J^*_0} = u_0$. The total number J^*_n of internal iterations may be explicitly fixed or depends on the evaluation of some residual. Therefore, it will usually depend on the time t^n, hence the index n in the notation. Convergence of the iterative scheme is obtained assuming that the norm of the matrix $\mathrm{Id} - D^{-1}A$ verifies

$$\rho := \|(\mathrm{Id} - D^{-1}A)\| < 1. \tag{2.2}$$

The actual implementation of the parallel algorithm thus combines this Jacobi (or Gauss Seidel) iterative procedure with the parareal algorithm introduced in (1.2). The number of internal iterations to achieve convergence now depends on n and k so we will denote them as $J^*_{n,k}$. Let $\mathcal{F}^{T_N}_{J^*,\Delta T}$ be the fine solver over $[T_N, T_{N+1}]$ if we perform these $J^*_{n,k}$ internal iterations and achieve convergence at every time step. With the new fine solver being defined this way, the plain parareal in time scheme (1.2) can be considered as being exact.

Instead of (the recommended) $J^*_{n,k}$, we propose, for the definition of the scalable balanced parareal method (SBPM) to perform only few internal iterations J at every time step with

$$J < \min_{\substack{k\in\{0,\dots,K-1\}\\ n\in\{1,\dots,\underline{\underline{\mathrm{n}}}\}}} J^*_{n,k},$$

Where K denotes the number of parareal iterations to achieve (1.3).

Provided that this limit on the number of internal iterations does not destroy the convergence of the outer (parallel) iterations, this will accelerate the computations in the parareal scheme for two reasons:

- We reduce the number of internal iterations at every parareal iteration.
- The total number of internal iterations performed by $\mathcal{F}^{T_N}_{J^*,\Delta T}$ varies with the time interval $[T_N, T_{N+1}]$ (because $J^*_{n,k}$ depends on n and k). Since, in practice, the fine propagations $\mathcal{F}^{T_N}_{J^*,\Delta T}$, $0 \leq N \leq \underline{N}-1$ are computed concurrently at every parareal iteration, the internal iterative process creates an imbalance in the tasks that no longer arises if we prescribe a constant number J of iterations at every time step.

The truncation yields a non converged version $\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}$ of $\mathcal{F}^{T_N}_{J^*,\Delta T}$, gives rise to the SBPM that reads :

$$\begin{cases} U^{N+1}_{k+1,J} = \mathcal{G}^{T_N}_{\Delta T}(U^N_{k+1,J}) + \widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J}) - \mathcal{G}^{T_N}_{\Delta T}(U^N_{k,J}), & k \geq 0, \quad 0 \leq N \leq \underline{N} - 1 \\ U^{N+1}_{0,J} = \mathcal{G}^{T_N}_{\Delta T}(U^N_{0,J}), & 0 \leq N \leq \underline{N} - 1 \\ U^0_{0,J} = U^0_0 = u_0 \end{cases} \tag{2.3}$$

We analyse in what follows, under which hypothesis the scheme SBPM converges similarly to (1.2) (i.e. in a similar number of parareal iterations).

3. Convergence analysis of the fully scalable balanced parareal scheme. The main result in this respect is given in Theorem 3.2. In order to introduce it, it is necessary to explain more in detail the procedure over each interval $[T_N, T_{N+1}]$ that allows to define what we have denoted as $\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}$. The following notation will be important to easily switch from the global framework over $[0,T]$ to the local one over each $[T_N, T_{N+1}]$. For any $n \in \{0,\dots,\underline{n}\}$, we denote by $t_n^N \coloneqq T_N + n\delta t$ the local time steps, where $t_0^N = T_N$ and $t_{\underline{n}}^N = T_{N+1}$. The approximated fine solution $\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J})$ at time T_{N+1} is obtained by solving for $n = 0,\dots,\underline{n}-1$ and $j = 1,\dots,J$

$$\begin{cases} u_k^{n+1,j} = (\mathrm{Id} - D^{-1}A)u_k^{n+1,j-1} + D^{-1}Bu_k^{n,J} + D^{-1}f^n, \\ u_k^{0,J} = U^N_{k,J}, \end{cases} \tag{3.1}$$

where $u_k^{n,j}$ denotes the approximation of $u(t_n^N)$ at the k-th parareal iteration and j-th internal iteration and we set

$$\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J}) = u_k^{\underline{n},J}. \tag{3.2}$$

Note that the scheme (3.1) is incomplete in the sense that it is necessary to define a starting guess $u_k^{n+1,0}$ to initialize each internal iterations. We propose to take either

$$\text{(Case I)} \quad \begin{cases} u_k^{n+1,0} = u_k^{n,J}, & \text{if } k = 0 \\ u_k^{n+1,0} = u_{k-1}^{n+1,J}, & \text{if } k \geq 1, \end{cases} \tag{3.3}$$

or

$$\text{(Case II)} \quad \begin{cases} u_k^{n+1,0} = u_k^{n,J}, & \text{if } k = 0 \\ u_k^{n+1,0} = u_{k-1}^{n+1,J} + u_k^{n,J} - u_{k-1}^{n,J}, & \text{if } k \geq 1. \end{cases} \tag{3.4}$$

An initialization of the form (3.3) (resp. (3.4)) will be denoted in the following as "first case" (resp. "second case"). Note that, in the first case, we take over the internal iterations at the point where they were stopped in the previous parareal iteration $k-1$. In addition to this, in the second case, we also better take the dynamics of the process into account through the terms $u_k^{n,J} - u_{k-1}^{n,J}$.

REMARK 3.1. *Note that due to the definition of the above starting guesses, the notation* (3.2) *is incomplete since actually* $\widetilde{\mathcal{F}}$ *also depends on quantities at the previous iteration* $k-1$*. For the sake of simplicity, we shall not explicitly write this dependance.*

Let us now turn to the convergence analysis of the SBPM (2.3). Theorem 3.2 shows that, under reasonable hypothesis, the error

$$\mathcal{E}^N_{k,J} = \|U^N_{k,J} - U^N\|$$

between the parareal solution $U^N_{k,J}$ and the sequential fine solution $U^N \coloneqq u^{N,\underline{n}}$ tends to zero for all $N \in \{0,\dots \underline{N}\}$ as the parareal iterations k tend to infinity.

THEOREM 3.2. *Assume that we have the following stability hypothesis on* $\mathcal{F}^t_\tau$, $\mathcal{G}^t_\tau$ *and* $\delta\mathcal{G}^t_\tau \coloneqq \mathcal{F}^t_\tau - \mathcal{G}^t_\tau$*:*

$$\|\delta\mathcal{G}^t_\tau(t,x)\| \leq C(\|x\|)\tau\varepsilon_{\delta\mathcal{G}} \tag{3.5a}$$

$$\|\mathcal{F}^t_\tau(t,x) - \mathcal{F}^t_\tau(t,y)\| \leq (1+C\tau)\|x-y\|, \tag{3.5b}$$

$$\|\mathcal{G}^t_\tau(t,x) - \mathcal{G}^t_\tau(t,y)\| \leq (1+C\tau)\|x-y\|, \tag{3.5c}$$

$$\|\delta\mathcal{G}^t_\tau(t,x) - \delta\mathcal{G}^t_\tau(t,y)\| \leq C\tau\varepsilon_{\delta\mathcal{G}}\|x-y\| \tag{3.5d}$$

Assume in addition that

$$\|\mathrm{Id} - A^{-1}B\| \leq C\delta t \tag{3.6}$$

and that

$$\rho^J \leq C\delta t \varepsilon_{\delta\mathcal{G}}^2, \quad \textit{in Case I}, \tag{3.7}$$

or

$$\rho^J \leq C \min\left(\delta t \varepsilon_{\delta\mathcal{G}}, \varepsilon_{\delta\mathcal{G}}^2\right), \quad \textit{in Case II}. \tag{3.8}$$

Then, there exists a constant $C > 0$ such that

$$\max_{N\in\{0,\dots,\underline{N}\}} \mathcal{E}_{k,J}^N \leq C\varepsilon_{\delta\mathcal{G}}^{k+1}, \quad k \geq 0$$

for the parareal algorithm (2.3) *with J internal iterations and with the first type of initialization* (3.3).

REMARK 3.3 (Remarks on the hypothesis of Theorem 3.2).

- *Hypothesis* (3.5a) *to* (3.5d) *are classical to prove the convergence of the plain parareal algorithm. In particular, note that* (3.5a) *and* (3.5d) *mean that $\mathcal{G}_\tau^t$ is $\varepsilon_{\delta\mathcal{G}}$ accurate with respect to $\mathcal{F}_\tau^t$.*
- *Hypothesis* (3.6) *is classical in the numerical analysis of the fine solver. Indeed, it allows to prove that the fine propagator is δt accurate. In addition, note that it implies that*

  $$\|A^{-1}B\| \leq 1 + C\delta t \tag{3.9}$$

  which is an inequality that will be used in the proof of the theorem. In a system of differential equations of the form of (1.1)*, the constant C will be of a fixed size that is independent of the spatial discretization. Finally, note that this property (3.6) can be weakened in the case where the system comes from the spatial discretization of a partial differential equation, like the heat equation. In this cases, $\mathcal{A}$ is symmetric positive definite and $A^{-1}B$ is a contraction, i.e., we have a better condition than* (3.9)

  $$\|A^{-1}B\| < 1.$$

We start with the proof of the Theorem in "Case I" for the initialization (3.3); the proof in that case requires the following intermediate result.

LEMMA 3.4. *With the hypothesis of Theorem 3.2, in "Case I", we have for $0 \leq N \leq \underline{N}$*

$$\begin{aligned} &\|\widetilde{\mathcal{F}}_{J,\Delta T}^{T_N}(U_{k,J}^N) - \mathcal{F}_{\Delta T}^{T_N}(U_{k,J}^N)\| \\ &\qquad \leq \begin{cases} C\Delta T\delta t\varepsilon_{\delta\mathcal{G}}^2, & k = 0, \\ C\Delta T\rho^J[\underline{n}\rho^J]^k + c\sum_{\ell=1}^k [\underline{n}\rho^J]^\ell[\mathcal{E}_{k-\ell,J}^N + \mathcal{E}_{k-\ell+1,J}^N], & k \geq 1. \end{cases} \end{aligned} \tag{3.10}$$

Proof. For any $k \geq 0$, the term $\mathcal{F}_{\Delta T}^{T_N}(U_{k,J}^N)$ is the solution $u_k^{\underline{\mathrm{n}}}$ to the following exact fine solver over $[T_N, T_{N+1}]$ given by

$$\begin{cases} u_k^{n+1} = A^{-1}Bu_k^n + A^{-1}f^n, \quad 0 \leq n \leq \underline{\mathrm{n}} - 1 \\ u_k^0 = U_{k,J}^N. \end{cases} \tag{3.11}$$

while the degraded fine propagator $\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J})$ is the solution $u^{\underline{\mathrm{n}},J}_k$ to problem (3.1) that we recall here : for $0 \le n \le \underline{n}-1$ and $1 \le j \le J$,

$$\begin{cases} u^{n+1,j}_k = (\mathrm{Id} - D^{-1}A)u^{n+1,j-1}_k + D^{-1}Bu^{n,J}_k + D^{-1}f^n, \\ u^{0,J}_k = U^N_{k,J}, \end{cases} \tag{3.12}$$

Let us now introduce an auxiliary notation $\widehat{u}^{\underline{\mathrm{n}}}_k$ defined through

$$\begin{cases} \widehat{u}^{n+1}_k = A^{-1}Bu^{n,J}_k + A^{-1}f^n, \quad 0 \le n \le \underline{n}-1, \\ \widehat{u}^0_k = U^N_{k,J}, \end{cases} \tag{3.13}$$

We now define the error sequences for $0 \le j \le J,\ 0 \le n \le \underline{\mathrm{n}},\ k \ge 0$,

$$\begin{cases} \tilde{e}^{n,j}_k = u^{n,j}_k - u^n_k, \\ \widehat{e}^n_k = \widehat{u}^n_k - u^n_k, \\ d^{n,j}_k = u^{n,j}_k - \widehat{u}^n_k = \tilde{e}^{n,j}_k - \widehat{e}^n_k. \end{cases}$$

Note that, with these notations, the term $\|\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J}) - \mathcal{F}^{T_N}_{\Delta T}(U^N_{k,J})\|$ that we want to bound reads $\|\tilde{e}^{\underline{\mathrm{n}},J}_k\|$. By subtracting (3.11) to (3.13), we infer that

$$\widehat{e}^{n+1}_k = A^{-1}B\tilde{e}^{n,J}_k, \quad 0 \le n \le \underline{\mathrm{n}}-1 \tag{3.14}$$

Similarly, subtracting (3.12) to (3.13), we can derive that $d^{n+1,j}_k = (\mathrm{Id} - D^{-1}A)d^{n+1,j-1}_k$, which with a bootstrap over j yields

$$d^{n+1,j}_k = (\mathrm{Id} - D^{-1}A)^J d^{n+1,0}_k \tag{3.15}$$

The term $d^{n+1,0}_k$ can be evaluated as follows

$$d^{n+1,0}_k = \tilde{e}^{n+1,0}_k - \widehat{e}^{n+1}_k = \tilde{e}^{n+1,0}_k - A^{-1}B\tilde{e}^{n,J}_k, \tag{3.16}$$

where we have used (3.14) to derive the last equality. For $k \ge 0,\ 0 \le n \le \underline{\mathrm{n}}-1$, we thus obtain

$$\tilde{e}^{n+1,J}_k = d^{n+1,J}_k + \widehat{e}^{n+1}_k = (\mathrm{Id} - D^{-1}A)^J\tilde{e}^{n+1,0}_k + [\mathrm{Id} - (\mathrm{Id} - D^{-1}A)^J]A^{-1}B\tilde{e}^{n,J}_k \tag{3.17}$$

which, thanks to (3.9) and (2.2) allows to bound $\|\tilde{e}^{n+1,J}_k\|$ by

$$\|\tilde{e}^{n+1,J}_k\| \le \rho^J\|\tilde{e}^{n+1,0}_k\| + (1+\rho^J)(1+C\delta t)\|\tilde{e}^{n,J}_k\|. \tag{3.18}$$

We now bound $\|\tilde{e}^{n+1,0}_k\| = \|u^{n+1,0}_k - u^{n+1}_k\|$ by using the first type of initialization of the iterations on j that was defined in (3.3). It can easily be seen that, in this case,

$$\|\tilde{e}^{n+1,0}_k\| \le \begin{cases} \|\tilde{e}^{n,J}_k\| + \|u^n_k - u^{n+1}_k\|, & k = 0, \\ \|\tilde{e}^{n+1,J}_{k-1}\| + \|u^{n+1}_{k-1} - u^{n+1}_k\|, & k \ge 1. \end{cases} \tag{3.19}$$

Note also that, by property (3.9), we have, on one hand,

$$\|u^n_k - u^{n+1}_k\| \le C\delta t, \quad k \ge 0 \tag{3.20}$$

and, on the other hand, we can bound $\|u_{k-1}^{n+1} - u_k^{n+1}\|$ as

$$\begin{aligned}\|u_{k-1}^{n+1} - u_k^{n+1}\| &= \|\mathcal{F}_{(n+1)\delta t}^{T_N}\left(U_{k-1,J}^N\right) - \mathcal{F}_{(n+1)\delta t}^{T_N}\left(U_{k,J}^N\right)\| \\ &\leq (1 + C(n+1)\delta t)\,\|U_{k-1,J}^N - U_{k,J}^N\| \\ &\leq (1 + C(n+1)\delta t)\,[\|U_{k-1,J}^N - U^N\| + \|U_{k,J}^N - U^N\|] \\ &= (1 + C\Delta T)\,[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N], \quad k \geq 1. \end{aligned} \tag{3.21}$$

By using (3.19), (3.20) and (3.21) in inequality (3.18), we infer

$$\|\tilde{e}_k^{n+1,J}\| \leq \begin{cases} C\delta t\rho^J + \theta\|\tilde{e}_k^{n,J}\|, & k = 0, \quad 0 \leq n \leq \underline{\mathrm{n}} - 1 \\ \rho^J \max\limits_{0\leq m\leq \underline{\mathrm{n}}}\|\tilde{e}_{k-1}^{m,J}\| + \rho^J\,(1 + C\Delta T)\,[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N] + \theta\|\tilde{e}_k^{n,J}\|, & k \geq 1, \quad 0 \leq n \leq \underline{\mathrm{n}} - 1, \end{cases}$$

where $\theta = (1 + C\delta t)(1 + 2\rho^J)$. A bootstrap over n yields (note that $\tilde{e}_k^{0,J} \equiv 0$)

$$\|\tilde{e}_k^{n+1,J}\| \leq \begin{cases} \dfrac{1-\theta^n}{1-\theta} C\delta t\rho^J, & k = 0, \quad 0 \leq n \leq \underline{\mathrm{n}} - 1 \\ \dfrac{1-\theta^n}{1-\theta}\rho^J\left(\max\limits_{0\leq m\leq \underline{\mathrm{n}}}\|\tilde{e}_{k-1}^{m,J}\| + (1 + C\Delta T)\,[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N]\right), & k \geq 1, \quad 0 \leq n \leq \underline{\mathrm{n}} - 1. \end{cases}$$

For $0 \leq n \leq \underline{\mathrm{n}}$, we have $\frac{1-\theta^n}{1-\theta} \leq n\theta^n \leq \underline{\mathrm{n}}\theta^{\underline{\mathrm{n}}}$. Besides, $\theta^{\underline{\mathrm{n}}} \leq C(1 + \underline{\mathrm{n}}\delta t)(1 + 2\underline{\mathrm{n}}\rho^J)$ with a moderate constant C since δt and ρ^J are small quantities. Since $\rho^J \leq C\delta t\varepsilon_{\delta\mathcal{G}}^2$, we have that $\underline{\mathrm{n}}\theta^{\underline{\mathrm{n}}} \leq C(1 + \Delta T)(1 + 2C\Delta T\varepsilon_{\delta\mathcal{G}}^2)$, which implies that

$$\frac{1-\theta^n}{1-\theta} \leq C\underline{\mathrm{n}}, \tag{3.22}$$

with a moderate constant C. Therefore, for $k = 0$,

$$\max_{0\leq m\leq \underline{\mathrm{n}}}\|\tilde{e}_{k=0}^{m,J}\| \leq C\Delta T\rho^J, \quad 0 \leq n \leq \underline{\mathrm{n}}, \tag{3.23}$$

which concludes the proof of the Lemma for $k = 0$ since $\|\widetilde{\mathcal{F}}_{J,\Delta T}^{T_N}(U_{k=0,J}^N) - \mathcal{F}_{\Delta T}^{T_N}(U_{k=0,J}^N)\| = \|\tilde{e}_{k=0}^{\underline{\mathrm{n}},J}\| \leq \max\limits_{0\leq m\leq \underline{\mathrm{n}}}\|\tilde{e}_{k=0}^{m,J}\|$. In the case $k \geq 1$, inequality (3.22) yields

$$\max_{0\leq m\leq \underline{\mathrm{n}}}\|\tilde{e}_k^{m,J}\| \leq C\underline{\mathrm{n}}\rho^J\left(\max_{0\leq m\leq \underline{\mathrm{n}}}\|\tilde{e}_{k-1}^{m,J}\| + (1 + C\Delta T)\,[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N]\right),$$

which, by a bootstrap argument over k gives the desired result

$$\max_{m,0\leq m\leq \underline{\mathrm{n}}}\|\tilde{e}_k^{m,J}\| \leq C\Delta T\rho^J[\underline{\mathrm{n}}\rho^J]^k + c\sum_{\ell=1}^{k}[\underline{\mathrm{n}}\rho^J]^\ell[\mathcal{E}_{k-\ell,J}^N + \mathcal{E}_{k-\ell+1,J}^N], \quad k \geq 1.$$

□

Proof. [Proof of Theorem 3.2 in "Case I" for the initialization (3.3)] We proceed by induction over k. In the case $k = 0$, $U_{k=0,J}^{N+1} = \mathcal{G}_{\Delta T}^{T_N}(U_{k=0,J}^N)$ for $0 \leq N \leq \underline{N} - 1$ and $U_{0,J}^0 = u_0$. Thus, for $0 \leq N \leq \underline{N} - 1$,

$$\begin{aligned}\mathcal{E}_{0,J}^{N+1} &= \|U_{0,J}^{N+1} - U^{N+1}\| \\ &= \|\mathcal{G}_{\Delta T}^{T_N}(U_{0,J}^N) - \mathcal{F}_{\Delta T}^{T_N}(U^N)\| \\ &\leq \|\mathcal{G}_{\Delta T}^{T_N}(U_{0,J}^N) - \mathcal{G}_{\Delta T}^{T_N}(U^N)\| + \|\mathcal{G}_{\Delta T}^{T_N}(U^N) - \mathcal{F}_{\Delta T}^{T_N}(U^N)\| \\ &\leq (1 + C\Delta T)\|U_{0,J}^N - U^N\| + C\Delta T\varepsilon_{\delta\mathcal{G}} = (1 + C\Delta T)\mathcal{E}_{0,J}^N + C\Delta T\varepsilon_{\delta\mathcal{G}}, \end{aligned}$$

where we have used hypothesis (3.5c) and (3.5a) to derive the last inequality. A bootstrap argument over N allows to derive the result

$$\max_{N\in\{0,\dots \underline{N}\}} \mathcal{E}_{0,J}^{N+1} \leq C\varepsilon_{\delta\mathcal{G}}. \tag{3.24}$$

For $k \geq 1$, we proceed as follows. From (2.3), we can write for $k \geq 0$

$$\begin{aligned} U_{k+1,J}^{N+1} - U^{N+1} &= \mathcal{G}_{\Delta T}^{T_N}(U_{k+1,J}^N) - \mathcal{G}_{\Delta T}^{T_N}(U^N) - \mathcal{G}_{\Delta T}^{T_N}(U_{k,J}^N) + \mathcal{G}_{\Delta T}^{T_N}(U^N) \\ &\quad + \widetilde{\mathcal{F}}_{J,\Delta T}^{T_N}(U_{k,J}^N) - \mathcal{F}_{\Delta T}^{T_N}(U^N) \\ &= \mathcal{G}_{\Delta T}^{T_N}(U_{k+1,J}^N) - \mathcal{G}_{\Delta T}^{T_N}(U^N) - [\mathcal{G}_{\Delta T}^{T_N} - \mathcal{F}_{\Delta T}^{T_N}](U_{k,J}^N) + [\mathcal{G}_{\Delta T}^{T_N} - \mathcal{F}_{\Delta T}^{T_N}](U^N) \\ &\quad + \widetilde{\mathcal{F}}_{J,\Delta T}^{T_N}(U_{k,J}^N) - \mathcal{F}_{\Delta T}^{T_N}(U_{k,J}^N). \end{aligned} \tag{3.25}$$

By taking norms in (3.25) and using (3.5c) and (3.5d), we derive for $k \geq 0$

$$\mathcal{E}_{k+1,J}^{N+1} \leq [1 + C\Delta T]\mathcal{E}_{k+1,J}^N + C\varepsilon_{\delta\mathcal{G}}\Delta T\mathcal{E}_{k,J}^N + \|\widetilde{\mathcal{F}}_{J,\Delta T}^{T_N}(U_{k,J}^N) - \mathcal{F}_{\Delta T}^{T_N}(U_{k,J}^N)\|. \tag{3.26}$$

By using the assumption that $\rho^J \leq C\delta t\varepsilon_{\delta\mathcal{G}}^2$ and the induction hypothesis

$$\max_{N\in\{0,\dots \underline{N}\}} \mathcal{E}_{p,J}^N \leq C\varepsilon_{\delta\mathcal{G}}^{p+1}, \quad 0 \leq p \leq k$$

in inequality (3.10), we derive that

$$\|\widetilde{\mathcal{F}}_{J,\Delta T}^{T_N}(U_{k,J}^N) - \mathcal{F}_{\Delta T}^{T_N}(U_{k,J}^N)\| \leq C\Delta T\varepsilon_{\delta\mathcal{G}}^{k+2}, \quad k \geq 0.$$

Using this inequality in (3.26) yields

$$\mathcal{E}_{k+1,J}^{N+1} \leq [1 + C\Delta T]\mathcal{E}_{k+1,J}^N + C\Delta T\varepsilon_{\delta\mathcal{G}}^{k+2}, \quad k \geq 0.$$

Finally, a bootstrap over N gives

$$\mathcal{E}_{k+1,J}^{N+1} \leq C\varepsilon_{\delta\mathcal{G}}^{k+2}, \quad 0 \leq N \leq \underline{\mathrm{N}} - 1.$$

□

We finish by the proof of the Theorem in "Case II" for the initialization (3.4). Here again we start with some preliminary lemmas

LEMMA 3.5. *With the hypothesis of Theorem 3.2, in "Case II", If $k = 0$ and $n = 1$,*

$$\|\tilde{e}_0^{1,0}\| \leq C\delta t. \tag{3.27}$$

In addition, if $k = 0$, we have

$$\max_{0\leq m\leq \underline{n}} \|\tilde{e}_0^{m,0}\| \leq C\Delta T\rho^J \tag{3.28}$$

If $k \geq 1$ and $n = 1$,

$$\|\tilde{e}_k^{1,0}\| \leq \max_{0\leq m\leq \underline{n}} \|\tilde{e}_{k-1}^{m,J}\| + C\delta t(1 + \Delta T)[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N] \tag{3.29}$$

In addition, for $k \geq 1$,

$$\max_{0\leq m\leq \underline{n}} \|\tilde{e}_k^{m,0}\| \leq \max_{0\leq m\leq \underline{n}} \|\tilde{e}_k^{m,J}\| + 2\max_{0\leq m\leq \underline{n}} \|\tilde{e}_{k-1}^{m,J}\| + C\delta t(1+\Delta T)[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N] \tag{3.30}$$

Also, for $k \geq 1,\ 0 \leq n \leq \underline{n} - 1$,

$$\|\alpha_k^n\| = \|u_k^{n+1} - u_k^n - u_{k-1}^{n+1} + u_{k-1}^n\| \leq C\delta t\,(1 + C\Delta T)\,[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N]. \tag{3.31}$$

Proof. The result for $k = 0$ was proven in the lemma for the first type of initialization. For $k \geq 1$ and $0 \leq n \leq \underline{\mathrm{n}} - 1$, we have with the second type of initialization,

$$\begin{aligned} \tilde{e}_k^{n+1,0} &= u_k^{n+1,0} - u_k^{n+1} \\ &= u_{k-1}^{n+1,J} + u_k^{n,J} - u_{k-1}^{n,J} - u_k^{n+1} \\ &= \tilde{e}_{k-1}^{n+1,J} + \tilde{e}_k^{n,J} - \tilde{e}_{k-1}^{n,J} - \alpha_k^n, \end{aligned} \tag{3.32}$$

where

$$\alpha_k^n := u_k^{n+1} - u_k^n - u_{k-1}^{n+1} + u_{k-1}^n, \quad k \geq 1,\ 0 \leq n \leq \underline{\mathrm{n}} - 1. \tag{3.33}$$

From (3.32) we derive

$$\|\tilde{e}_k^{n+1,0}\| \leq \max_{0 \leq m \leq \underline{n}} \|\tilde{e}_k^{m,J}\| + 2 \max_{0 \leq m \leq \underline{n}} \|\tilde{e}_{k-1}^{m,J}\| + \|\alpha_k^n\|. \tag{3.34}$$

Let us bound $\|\alpha_k^n\|$. For this, it is first all immediate from (3.11) that

$$u_k^{n+1} - u_{k-1}^{n+1} = A^{-1}B[u_k^n - u_{k-1}^n],$$

from which we derive

$$u_k^{n+1} - u_k^n - u_{k-1}^{n+1} + u_{k-1}^n = -(\mathrm{Id} - A^{-1}B)[u_k^n - u_{k-1}^n] = -(\mathrm{Id} - A^{-1}B)[A^{-1}B]^n[u_k^0 - u_{k-1}^0]$$

By using (3.6) in the last formula, for $k \geq 1$ and $0 \leq n \leq \underline{\mathrm{n}} - 1$ we get,

$$\|\alpha_k^n\| = \|u_k^{n+1} - u_k^n - u_{k-1}^{n+1} + u_{k-1}^n\| \leq C\delta t\,(1 + Cn\delta t)\,\|U_{k,J}^N - U_{k-1,J}^N\|, \tag{3.35}$$

where again (3.11) has been used hence, improving (3.20), by a factor δt, indeed

$$\|\alpha_k^n\| = \|u_k^{n+1} - u_k^n - u_{k-1}^{n+1} + u_{k-1}^n\| \leq C\delta t\,(1 + C\Delta T)\,[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N], \quad k \geq 1. \tag{3.36}$$

When $k \geq 1$ and $n = 0$, relation (3.32) reads

$$\tilde{e}_k^{1,0} = \tilde{e}_{k-1}^{1,J} - \alpha_k^0, \tag{3.37}$$

and easily yields the desired bound by using (3.36). □ We then need the following result for the iterations $k \geq 1$ the proof of which is straightforward

LEMMA 3.6. *Let us set* $\mathcal{A} := \mathcal{B} + (\mathrm{Id} - \mathcal{B})A^{-1}B$, $\mathcal{B} := (\mathrm{Id} - D^{-1}A)^J$ *and* $K := \sum_{p=0}^{J-1}(\mathrm{Id} - D^{-1}A)^p D^{-1}B$ *then*

$$\begin{cases} \|\mathcal{A}\|^n & \leq 1 + n\delta t \\ \|\mathcal{B}\| & \leq \rho^J \\ \|K\| & \leq (1 + C\delta t) \end{cases} \tag{3.38}$$

LEMMA 3.7. *If, for* $k \geq 0$,

$$\mathcal{E}_{k,J}^N \leq \varepsilon_{\delta\mathcal{G}}^{k+1}, \quad 0 \leq N \leq \underline{N}, \tag{3.39}$$

then, with the hypothesis of Theorem 3.2 for Case II, we have for $k \geq 0$*,*

$$\|\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J}) - \mathcal{F}^{T_N}_{\Delta T}(U^N_{k,J})\| \leq \varepsilon^{k+2}_{\delta\mathcal{G}}, \quad 0 \leq N \leq \underline{N}. \tag{3.40}$$

Proof. For any $k \geq 0$, the term $\mathcal{F}^{T_N}_{\Delta T}(U^N_{k,J})$ is the solution $u^{\underline{\mathrm{n}}}_k$ to the following exact fine solver over $[T_N, T_{N+1}]$ given by

$$\begin{cases} u^{n+1}_k = A^{-1}Bu^n_k + A^{-1}f^n, \quad 0 \leq n \leq \underline{\mathrm{n}} - 1 \\ u^0_k = U^N_{k,J}. \end{cases} \tag{3.41}$$

while the degraded fine propagator $\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J})$ is the solution $u^{\underline{\mathrm{n}},J}_k$ to problem (3.1) that we recall here: for $0 \leq n \leq \underline{n} - 1$ and $1 \leq j \leq J$,

$$\begin{cases} u^{n+1,j}_k = (\mathrm{Id} - D^{-1}A)u^{n+1,j-1}_k + D^{-1}Bu^{n,J}_k + D^{-1}f^n, \\ u^{0,J}_k = U^N_{k,J}, \end{cases} \tag{3.42}$$

We now define the error sequence

$$\tilde{e}^{n,j}_k := u^{n,j}_k - u^n_k, \quad 0 \leq j \leq J,\ 0 \leq n \leq \underline{\mathrm{n}},\ k \geq 0,$$

where, by definition, $\tilde{e}^{0,j}_k = 0$. With these notations, the term $\|\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J}) - \mathcal{F}^{T_N}_{\Delta T}(U^N_{k,J})\|$ that we want to bound reads $\|\tilde{e}^{\underline{\mathrm{n}},J}_k\|$. For $k \geq 0,\ 0 \leq n \leq \underline{\mathrm{n}} - 1$, we have derived in (3.17)

$$\tilde{e}^{n+1,J}_k = (\mathrm{Id} - D^{-1}A)^J\tilde{e}^{n+1,0}_k + [\mathrm{Id} - (\mathrm{Id} - D^{-1}A)^J]A^{-1}B\tilde{e}^{n,J}_k,$$

which, by using the second type of initialization (3.4) can be written for $k \geq 1$ as

$$\tilde{e}^{n+1,J}_k = \left(\mathcal{B} + (\mathrm{Id} - \mathcal{B})A^{-1}B\right)\tilde{e}^{n,J}_k + \mathcal{B}\left(\tilde{e}^{n+1,J}_{k-1} - \tilde{e}^{n,J}_{k-1}\right) - \mathcal{B}\alpha^n_k, \quad 0 \leq n \leq \underline{\mathrm{n}} - 1, \tag{3.43}$$

where $\alpha^n_k := u^{n+1}_k - u^n_k - u^{n+1}_{k-1} + u^n_{k-1}$. This expression motivates the analysis of the expression

$$\tilde{e}^{n+1,J}_{k-1} - \tilde{e}^{n,J}_{k-1} = u^{n+1,J}_{k-1} - u^{n+1}_{k-1} - u^{n,J}_{k-1} + u^n_{k-1}, \quad k \geq 1,\ 0 \leq n \leq \underline{\mathrm{n}} - 1. \tag{3.44}$$

For this, we first notice that from (3.41) we obtain, for $1 \leq n \leq \underline{\mathrm{n}} - 1$

$$u^{n+1}_{k-1} - u^n_{k-1} = A^{-1}\left(B[u^n_{k-1} - u^{n-1}_{k-1}] + [f^n - f^{n-1}]\right),$$

then, using the fact that $\rho = \|\mathrm{Id} - D^{-1}A\| < 1$, we derive $A^{-1}D = \sum_{p=0}^{\infty}(\mathrm{Id} - D^{-1}A)^p$, hence

$$u^{n+1}_{k-1} - u^n_{k-1} = \sum_{p=0}^{\infty}(\mathrm{Id} - D^{-1}A)^pD^{-1}\left(B[u^n_{k-1} - u^{n-1}_{k-1}] + [f^n - f^{n-1}]\right) \tag{3.45}$$

for $1 \leq n \leq \underline{\mathrm{n}} - 1$. Equation (3.45) rewrites

$$\begin{aligned} u_{k-1}^{n+1} - u_{k-1}^{n} &= (\mathrm{Id} - D^{-1}A)^J \sum_{p=0}^{\infty} (\mathrm{Id} - D^{-1}A)^p D^{-1} \left(B[u_{k-1}^n - u_{k-1}^{n-1}] + [f^n - f^{n-1}] \right) \\ &\quad + \sum_{p=0}^{J-1} (\mathrm{Id} - D^{-1}A)^p D^{-1} \left(B[u_{k-1}^n - u_{k-1}^{n-1}] + [f^n - f^{n-1}] \right) \\ &= (\mathrm{Id} - D^{-1}A)^J [u_{k-1}^{n+1} - u_{k-1}^{n}] \\ &\quad + \left(\sum_{p=0}^{J-1} (\mathrm{Id} - D^{-1}A)^p D^{-1} \right) \left(B[u_{k-1}^n - u_{k-1}^{n-1}] + [f^n - f^{n-1}] \right). \end{aligned}$$

On the other hand, (3.42) leads to

$$\begin{aligned} u_{k-1}^{n+1,J} - u_{k-1}^{n,J} &= (\mathrm{Id} - D^{-1}A)^J [u_{k-1}^{n+1,0} - u_{k-1}^{n,0}] \\ &\quad + \left(\sum_{p=0}^{J-1} (\mathrm{Id} - D^{-1}A)^p D^{-1} \right) \left(B[u_{k-1}^{n,J} - u_{k-1}^{n-1,J}] + [f^n - f^{n-1}] \right), \end{aligned}$$

for $1 \leq n \leq \underline{\mathrm{n}} - 1$. By subtraction, we get for $k \geq 1$ and $1 \leq n \leq \underline{\mathrm{n}} - 1$,

$$\begin{aligned} \tilde{e}_{k-1}^{n+1,J} - \tilde{e}_{k-1}^{n,J} &= (\mathrm{Id} - D^{-1}A)^J [\tilde{e}_{k-1}^{n+1,0} - \tilde{e}_{k-1}^{n,0}] \\ &\quad + \left(\sum_{p=0}^{J-1} (\mathrm{Id} - D^{-1}A)^p D^{-1} B \right) (\tilde{e}_{k-1}^{n,J} - \tilde{e}_{k-1}^{n-1,J}). \end{aligned}$$

A bootstrap argument over n yields for $k \geq 1$ and $1 \leq n \leq \underline{\mathrm{n}} - 1$,

$$\tilde{e}_{k-1}^{n+1,J} - \tilde{e}_{k-1}^{n,J} = \sum_{\ell=1}^{n} K^{n-\ell} \mathcal{B} \left(\tilde{e}_{k-1}^{\ell+1,0} - \tilde{e}_{k-1}^{\ell,0} \right) + K^n \left(\tilde{e}_{k-1}^{1,0} - \tilde{e}_{k-1}^{0,0} \right), \tag{3.46}$$

where we note that, in fact, $\tilde{e}_{k-1}^{0,0} = 0$ by definition. By inserting (3.46) in (3.43) and by rearranging terms, we get for $1 \leq n \leq \underline{\mathrm{n}} - 1,\ k \geq 1$,

$$\tilde{e}_{k}^{n+1,J} = \mathcal{A}\tilde{e}_{k}^{n,J} - \mathcal{B}\alpha_k^n + \mathcal{B}v_{k-1}^n + \mathcal{B}K^n \tilde{e}_{k-1}^{1,0}, \tag{3.47}$$

where $v_{k-1}^n \coloneqq \sum_{m=1}^{n} K^{n-m} \mathcal{B}(\tilde{e}_{k-1}^{m+1,0} - \tilde{e}_{k-1}^{m,0})$. A bootstrap over n yields for $1 \leq n \leq \underline{\mathrm{n}} - 1,\ k \geq 1$,

$$\tilde{e}_{k}^{n+1,J} = \mathcal{A}\tilde{e}_{k}^{1,J} - \sum_{\ell=1}^{n} \mathcal{A}^{n-\ell}\mathcal{B}\alpha_k^\ell + \sum_{\ell=1}^{n} \mathcal{A}^{n-\ell}\mathcal{B}^2 v_{k-1}^\ell + \left(\sum_{\ell=1}^{n} \mathcal{A}^{n-\ell}\mathcal{B}K^\ell \right) \tilde{e}_{k-1}^{1,J}. \tag{3.48}$$

By taking norms in the last expression, we can derive the following bound for $\|\tilde{e}_{k}^{n+1,J}\|$,

$$\begin{aligned} \|\tilde{e}_{k}^{n+1,J}\| &\leq \|\mathcal{A}\|^n \|\tilde{e}_{k}^{1,J}\| + \underline{\mathrm{n}} \|\mathcal{B}\| \max_{1 \leq \ell \leq \underline{\mathrm{n}}} \|\mathcal{A}\|^{\underline{\mathrm{n}}-\ell} \max_{1 \leq \ell \leq \underline{\mathrm{n}}} \|\alpha_k^\ell\| \\ &\quad + \underline{\mathrm{n}} \|\mathcal{B}\| \max_{1 \leq \ell \leq \underline{\mathrm{n}}} \|\mathcal{A}\|^{\underline{\mathrm{n}}-\ell} \max_{1 \leq \ell \leq \underline{\mathrm{n}}} \|v_{k-1}^\ell\| \\ &\quad + \underline{\mathrm{n}} \|\mathcal{B}\| \|\tilde{e}_{k-1}^{1,J}\| \max_{1 \leq \ell \leq \underline{\mathrm{n}}} \|\mathcal{A}\|^{\underline{\mathrm{n}}-\ell} \max_{1 \leq \ell \leq \underline{\mathrm{n}}} \|K\|^\ell, \quad 1 \leq n \leq \underline{\mathrm{n}} - 1,\ k \geq 1, \end{aligned} \tag{3.49}$$

from which we shall derive the desired result by bounding recursively $\max_{0\le m\le \underline{\mathrm{n}}}\|\tilde{e}_k^{m,J}\|$ for $k\ge 1$. The case $k=0$ has been treated in the previous theorem for the first initialization. The result is valid here and reads,

$$\max_{0\le m\le \underline{\mathrm{n}}}\|\tilde{e}_0^{m,J}\| \le C\Delta T\rho^J \le C\Delta T\varepsilon_{\delta\mathcal{G}}^2, \tag{3.50}$$

where the last inequality comes from the fact that, by hypothesis here, $\rho^J \le C\min\left(\delta t\varepsilon_{\delta\mathcal{G}}, \varepsilon_{\delta\mathcal{G}}^2\right)$. In addition, for $k\ge 0$,

(3.51)
$$\|\tilde{e}_k^{1,J}\| \le \rho^J\|\tilde{e}_k^{1,0}\| \le \begin{cases} C\rho^J\delta t, \text{ if } k=0 \\ C\rho^J\left(\max_{0\le m\le \underline{\mathrm{n}}}\|\tilde{e}_{k-1}^{m,J}\| + C\delta t(1+\Delta T)[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N]\right), \text{ if } k\ge 1 \end{cases}$$

where we have used (3.27) and (3.29) to bound $\|\tilde{e}_k^{1,0}\|$.

Finally, since $\max_{1\le\ell\le\underline{\mathrm{n}}}\|v_k^\ell\| \le 2\underline{\mathrm{n}}\|\mathcal{B}\|\|K\|^{\underline{\mathrm{n}}}\max_{1\le\ell\le\underline{\mathrm{n}}}\|\tilde{e}_k^{\ell,0}\|$, then, if $k=0$, using (3.28) and (3.38) we derive

$$\max_{1\le\ell\le\underline{\mathrm{n}}}\|v_0^\ell\| \le C\underline{\mathrm{n}}\rho^J(1+\Delta T)\Delta T\rho^J. \tag{3.52}$$

If $k\ge 1$, using (3.29) and (3.38),

(3.53)
$$\max_{1\le\ell\le\underline{\mathrm{n}}}\|v_k^\ell\| \le C\underline{\mathrm{n}}(1+\Delta T)\left(\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_k^{m,J}\| + 2\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_{k-1}^{m,J}\| + C\delta t(1+\Delta T)[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N]\right).$$

Let us come back to the bound for $\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_k^{m,J}\|$. If $k\ge 1$, by bounding the members of the right hand side of (3.49) with (3.38), (3.51), (3.31), (3.52), we derive

$$\begin{aligned}\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_1^{m,J}\| \le{}& C(1+\Delta T)\rho^J\left(\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_0^{m,J}\| + C\delta t(1+\Delta T)[\mathcal{E}_{1,J}^N + \mathcal{E}_{0,J}^N]\right)\\ &+ C\underline{\mathrm{n}}\rho^J(1+\Delta T)\delta t(1+C\Delta T)[\mathcal{E}_{1,J}^N + \mathcal{E}_{0,J}^N]\\ &+ C\underline{\mathrm{n}}\rho^{2J}(1+\Delta T)\underline{\mathrm{n}}(1+\Delta T)\Delta T\rho^J\\ &+ C\underline{\mathrm{n}}\rho^J\rho^J\delta t(1+\Delta T)^2,\end{aligned}$$

which gives the desired result

$$\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_1^{m,J}\| \le C\varepsilon_{\delta\mathcal{G}}^3 \tag{3.54}$$

thanks to the bound (3.8) on ρ^J, hypothesis (3.39) on $\mathcal{E}_{k,J}^N$ and the bound (3.28) for $\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_0^{m,J}\|$. We now derive the result for $k\ge 2$ by induction where the recurrence hypothesis is $\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_p^{m,J}\| \le C\varepsilon_{\delta\mathcal{G}}^{p+2}$, $0\le p\le k-1$. In a similar manner as we have just derived in the case $k=1$, for any $k\ge 2$ we have

$$\begin{aligned}&\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_k^{m,J}\|\\ &\le C(1+\Delta T)\rho^J\left(\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_k^{m,J}\| + C\delta t(1+\Delta T)[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N]\right)\\ &+ \underline{\mathrm{n}}\rho^J(1+\Delta T)\delta t\,(1+C\Delta T)\,[\mathcal{E}_{k,J}^N + \mathcal{E}_{k-1,J}^N]\\ &+ (\underline{\mathrm{n}}\rho^J)^2(1+\Delta T)^2\left(\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_{k-1}^{m,J}\| + 2\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_{k-2}^{m,J}\| + C\delta t(1+\Delta T)[\mathcal{E}_{k-1,J}^N + \mathcal{E}_{k-2,J}^N]\right)\\ &+ \underline{\mathrm{n}}\rho^J(1+\Delta T)^2\rho^J\left(\max_{0\le m\le\underline{\mathrm{n}}}\|\tilde{e}_{k-2}^{m,J}\| + C\delta t(1+\Delta T)[\mathcal{E}_{k-1,J}^N + \mathcal{E}_{k-2,J}^N]\right),\end{aligned}$$

from which we infer that

$$\max_{0\leq m\leq \underline{\mathrm{n}}} \|\tilde{e}_k^{m,J}\| \leq \varepsilon_{\delta\mathcal{G}}^{k+2} \tag{3.55}$$

thanks to the recurrence hypothesis, the bound (3.8) on ρ^J and (3.39) on $\mathcal{E}_{k,J}^N$. □

Then the proof of the theorem in Case II proceeds exactly as in the first case.

4. Mitigation of potential memory storage issues. In the above described parareal scheme, the initialization of the internal iterations with (3.3) or (3.4) requires the knowledge of the fine states $u_{k-1}^{n,J}$ at all times t_n^N $(n = 0, \ldots, \underline{\mathrm{n}})$ from the previous iteration $k-1$. From an implementation point of view, given that the computation of each interval $[T_N, T_{N+1}]$ is performed by different processors, this implies that each processor must store the previous $\underline{\mathrm{n}}$ fine solutions corresponding to its time interval. This point can easily become an important bottleneck given the high memory demand that the storage of all these states can represent in many cases. An example is when the problem comes from the discretization of a PDE : each state has a size $\mathcal{N}$ equal to the number of degrees of freedom in the space direction, i.e. the size of the finite element mesh if such a discretization has been chosen.

We propose to address this issue by complexity reduction techniques : over every interval $[T_N, T_{N+1}]$, we propose to store the surrogates

$$\pi_M \left[u_{k-1}^{n,J}\right], \quad n = 0, \ldots, \underline{\mathrm{n}} \tag{4.1}$$

instead of the elements

$$u_{k-1}^{n,J}, \quad n = 0, \ldots, \underline{\mathrm{n}} \tag{4.2}$$

where π_M is the projection operator over an appropriate space X_M of dimension M much smaller than $\underline{\mathrm{n}}$. This transforms the previous $\underline{\mathrm{n}} \times \mathcal{N}$ storage requirement into a $M \times \mathcal{N} + \underline{\mathrm{n}}M$, since for every n, $n = 0, \ldots, \underline{\mathrm{n}}$, we store the M coefficients of each $u_{k-1}^{n,J}$ in a given basis of X_M and the knowledge of the basis requires the storage of $M \times \mathcal{N}$ data.

We define

$$\varepsilon_{\text{compr}} \coloneqq \max_{0\leq n\leq \underline{\mathrm{n}}} \|u_{k-1}^{n,J} - \pi_M \left[u_{k-1}^{n,J}\right]\|$$

as the error due to the "compression" procedure. The "perturbed" initializations read in this case

$$\text{(Case I)} \quad \begin{cases} u_k^{n+1,0} = u_k^{n,J}, & \text{if } k = 0, \\ u_k^{n+1,0} = \pi_M \left[u_{k-1}^{n+1,J}\right], & \text{if } k \geq 1, \end{cases} \tag{4.3}$$

or

$$\text{(Case II)} \quad \begin{cases} u_k^{n+1,0} = u_k^{n,J}, & \text{if } k = 0, \\ u_k^{n+1,0} = \pi_M \left[u_{k-1}^{n+1,J}\right] + u_k^{n,J} - \pi_M \left[u_{k-1}^{n,J}\right], & \text{if } k \geq 1. \end{cases} \tag{4.4}$$

This raises the questions:

1. How to build the reduced basis X_M without storing all the fine elements?
2. What is the smallest dimension M that allows to have good convergence properties in X_M for our standards? Is this dimension compatible with our storage limitations?

3. Are the convergence properties of our parareal scheme degraded with the use of the surrogates (4.1)? To what extend?

In the present paper, the answer to these questions will not be provided with theoretical arguments but only through numerical results (see section 5.3). Nevertheless, this enters in the frame of Theorem 3.2 since this additional error can be treated in the non converged part of the error (3.3).

Regarding the first and second points, a first approach could be to use some collocation method. For example, one could store a small set of solutions at different times and then interpolate to obtain the surrogates at other times. In our numerical simulations, another idea related to the construction of model reduction has been tested: assume that the dimension M is fixed *a priori* by our memory limitations. During the process of storing all the solutions $\{u_{k-1}^{n,J}\}_n$ in (4.2), we compress them regularly by using a Proper Orthogonal Decomposition (POD), and we denote by X_M the regularly updated reduced basis spanned by the largest M modes. Note that we do not wait to have the complete knowledge of all the fine solutions (4.2) that would imply storing the full set of solutions, we propose instead to build X_M through what could be called a "moving-window POD". Let us explain the procedure through an example: assume that $M = 10$ and that we can store a maximum number of $M_{\max} = 20$ elements. Let us fix $\underline{\mathrm{n}} = 100$ (then $n = 0, \ldots, 100$). We start by storing the first 20 fine solutions ($n = 0, \ldots, 19$) and extract out of these 20 modes a first POD basis $X_{10}^{(1)}$ of dimension 10. We now have in memory 10 elements so we continue by storing the next 10 fine solutions ($n = 20, \ldots, 29$). We now perform a second POD with the 10 basis functions of $X_{10}^{(1)}$ and the 10 new fine solutions. This gives $X_{10}^{(2)}$. And we continue the process.

REMARK 4.1. *Note that the construction of the reduced basis X_M and the "compression" of the fine solutions into X_M will slightly degrade the parallel efficiency of the scheme and its impact should be evaluated. A more detailed study about this particular point and also on how to determine their dimension M on the run will be analyzed in future works.*

5. An application to neutronics.

5.1. Overview of the general context and goals of our study. Accurate simulations of the evolution of the population of free neutrons in a reactor core $\mathcal{R}$ are critical in nuclear safety calculations, since the collision of free neutrons with fuel parts of the core leads to a chain of fission reactions that has to be kept under control to ensure the safety of the process. The concentration of free neutrons is usually modeled through an associated quantity called angular flux ψ which is governed by a linear Boltzmann equation whose terms physically express a balance between the free neutrons that are created and those that disappear in the core. We will place ourselves in the three dimensional case where ψ depends on seven variables, namely the time $t \in [0, T]$, the position within the reactor $\boldsymbol{r} \in \mathcal{R} \subset \mathbb{R}^3$ and the velocity of the neutrons $\boldsymbol{v} = \sqrt{2E/m}\boldsymbol{\omega}$, where $E \in [E_{\min}, E_{\max}]$ stands for the energy of the neutron, $\boldsymbol{\omega} = \boldsymbol{v}/|\boldsymbol{v}| \in \mathbb{S}_2$ stands for the direction of the velocity and m is the mass of the neutron. This is a highly complex problem that requires a lot of simulations. In spite of the use of some standard acceleration techniques employed in nuclear solvers (see, e.g., [9] and [14, Chapter 1]), the computing times for the resolution of this equation on realistic core geometries still remain too long with respect to the nuclear industry requirements. Hence our interest in exploring "less conventional" acceleration strategies like the parallelization of the time variable.

The ability of the parareal in time algorithm to speed up simulations based on these kinetic transport equations has been illustrated in [9] . As was explained above (and is quite classical, see, e.g., [1]), the scalability of the approach is rather low which is actually the motivation of this paper.

As a first step, instead of studying the evolution of ψ, we consider in the present paper, the evolution of the so-called scalar flux $\phi(t,\boldsymbol{r},E) \coloneqq \int_{\mathbb{S}_2} \psi(t,\boldsymbol{r},\boldsymbol{\omega},E)\mathrm{d}\boldsymbol{\omega'}$, which is the average of ψ over the angular variable. It is well-known (see, e.g. [4, Chapter XXI]) that ϕ is the solution of a parabolic equation which is less computationally expensive than the original Bolzmann equation. This simplification is actually very widespread in the field of nuclear calculations because it leads to accurate enough results in most of the usual cases on outputs like the generated power.

Another simplification that is traditionally done consists in averaging in the energy variable. This further approximation, known as the multi-group approach, is based on the division of the energy interval into G subintervals ($[E_{\min}, E_{\max}] = [E_G, E_{G-1}] \cup \cdots \cup [E_1, E_0]$) and leads to consider the set $\Phi = \{\phi^g\}_{g\in\{1,G\}}$ as the new unknown solution (see, e.g., [7, Chapter 3] or [4, Chapter XXI]).

Finally, the fission chain reaction that takes place inside the core leads to the presence of some radioactive isotopes that emit neutrons with a given delay (we refer to them as precursors of delayed neutrons). This phenomenon must be taken into account in the balance equation, hence the coupling of the parabolic equation for ϕ with a set of first order ODE's expressing the evolution in $\mathcal{R}$ of the precursors' concentration that will be denoted as $\mathbf{C} = \{C_\ell\}_{\ell\in\{1,\dots,L\}}$. As a result, the problem consists in finding for all $(t,\boldsymbol{r}) \in [0,T]\times\mathcal{R}$ the set of multigroup fluxes $\boldsymbol{\phi(t,r)} = \left(\phi^1(t,\boldsymbol{r}),\dots,\phi^G(t,\boldsymbol{r})\right)^T$ and the set of precursors' concentrations $\boldsymbol{C(t,r)} = \left(C_1(t,\boldsymbol{r}),\dots,C_L(t,\boldsymbol{r})\right)^T$ that are the solution of:

$$
\left\{
\begin{array}{l}
\dfrac{1}{V^g}\partial_t\phi^g(t,\boldsymbol{r}) - \nabla.\left[D^g(t,\boldsymbol{r})\nabla\phi^g(t,\boldsymbol{r})\right] + \sigma_t^g(t,\boldsymbol{r})\phi^g(t,\boldsymbol{r}) \\
\qquad\qquad - \displaystyle\sum_{g'=1}^{G} \sigma_s^{g'\to g}(t,\boldsymbol{r})\phi^{g'}(t,\boldsymbol{r}) \\
\qquad\qquad -\chi_p^g(t,\boldsymbol{r}) \displaystyle\sum_{g'=1}^{G} \Big(1-\beta^{g'}(t,\boldsymbol{r})\Big)\,(\nu\sigma_f)^{g'}(t,\boldsymbol{r})\phi^{g'}(t,\boldsymbol{r}) \\
\qquad\qquad - \displaystyle\sum_{\ell=1}^{L} \lambda_\ell\chi_{d,\ell}^g(t,\boldsymbol{r})C_\ell(t,\boldsymbol{r}) = 0,\ \forall g\in\{1,\dots,G\} \\
\partial_t C_\ell(t,\boldsymbol{r}) = -\lambda_\ell C_\ell(t,\boldsymbol{r}) \\
\qquad\qquad + \displaystyle\sum_{g'=1}^{G} \beta_\ell^{g'}(t,\boldsymbol{r})(\nu\sigma_f)^{g'}(t,\boldsymbol{r})\phi^{g'}(t,\boldsymbol{r}),\ \forall\ell\in\{1,\dots,L\}. \\
\phi^g(t,\boldsymbol{r}) = 0,\ \forall(t,\boldsymbol{r})\in[0,T]\times\partial\mathcal{R}.
\end{array}
\right. \tag{5.1}
$$

The initial conditions $\boldsymbol{\phi(0,r)}$ and $\boldsymbol{C(0,r)}$ are generally given by the resolution of the stationary diffusion equation. The coefficient V^g is the neutron velocity, D^g is the diffusion coefficient, σ_t^g is the total cross-section and $\sigma_s^{g'\to g}$ is the scattering cross-section from energy group g' to energy group g. The coefficients χ_p^g and $\chi_{d,\ell}^g$ are respectively the prompt spectrum in energy group g and the delayed spectrum of precursor ℓ in energy g. Finally, the terms λ_ℓ and β_ℓ^g are respectively the decay constant and the delayed neutron fraction of precursor ℓ at energy g.

Our computations will be done with a solver implemented in Freefem++ whose properties and accuracy have been presented in [3] for this diffusion problem. The

validation of the code and the parareal implementation is based on the so-called TWIGL benchmark [8], which is a well-known test case in the field of neutronics. It represents a rod withdrawal in a three-dimensional core

$$\mathcal{R} = \{(x, y, z) \in \mathbb{R}^3 |\ 0 \leq x \leq 220 \text{ cm};\ 0 \leq y \leq 220 \text{ cm};\ 0 \leq z \leq 200 \text{ cm}\}.$$

A cross-sectional view at the height $z = 180$ cm is specified in Figure 1. The first group of rods (blue) is withdrawn from $t = 0$ ($z = 100$ cm measured starting from below) until $t = 26.6$ s. ($z = 180$ cm) at a constant speed (see Figure 1). The second group of rods (red) is inserted from $t = 7.5$ s. ($z = 180$ cm) until $t = 47.7$ s. ($z = 60$ cm) the simulation was performed up to final time $T = 70$ s.

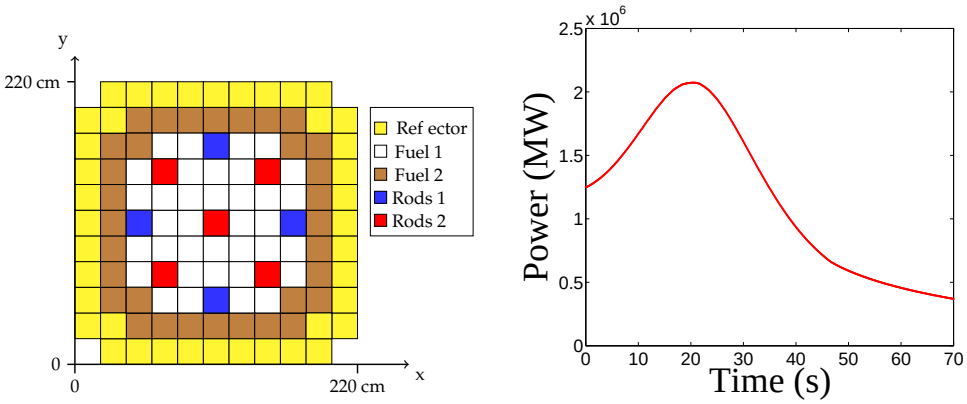

Figure 1: TWIGL benchmark. On the left: Cross-sectional view of the core at the height $z = 180$ *cm*. On the right: power evolution computed with our solver.

In what follows, we provide a preliminary study on the speed-up of the method by analyzing the number of fine internal iterations that are performed in each of the choices of starting guesses (case I or II). Finally, we also present here some first results concerning the impact of the use of surrogates for the internal starting guesses with a moving-window POD basis.

An extension of these results to the "full case" (Boltzmann equation + industrial solver) will be provided on a sequel of this paper.

5.2. Discretization of the model and application of the parareal algorithm with a degraded fine solver. For the resolution of (5.1), we define a fine solver that is built by applying an Euler backward discretization with time step δt. We denote by $\boldsymbol{\phi^n(r)} = \left(\phi^{1,n}(\boldsymbol{r}), \ldots, \phi^{G,n}(\boldsymbol{r})\right)^T$ and $\boldsymbol{C^n(r)} = (C_1^n(\boldsymbol{r}), \ldots, C_L^n(\boldsymbol{r}))^T$ the approximation of $\boldsymbol{\phi(t, r)}$ and $\boldsymbol{C(t, r)}$ at time $t = t^n$ with this fine solver, $n = 0, \ldots, \underline{\underline{\mathrm{n}}}$. At each time step, we are led to the resolution of a system that can be written in a block form:

Given $\boldsymbol{\phi^n}$ and $\boldsymbol{C^n}$, find $\boldsymbol{\phi^{n+1}}$ and $\boldsymbol{C^{n+1}}$ such that:

$$\begin{pmatrix} A_{1,1}^{n+1} & A_{1,2}^{n+1} \\ A_{2,1}^{n+1} & A_{2,2} \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi^{n+1}} \\ \boldsymbol{C^{n+1}} \end{pmatrix} = \frac{1}{\delta t} \begin{pmatrix} \boldsymbol{\phi^n} \\ \boldsymbol{C^n} \end{pmatrix}, \quad n = 0, \ldots, \underline{\underline{\mathrm{n}}} - 1. \tag{5.2}$$

The $G \times G$ operator matrix

$$A_{1,1}^{n+1} = \begin{pmatrix} a_{1,1}^{n+1} & \dots & a_{1,G}^{n+1} \\ \vdots & \ddots & \vdots \\ a_{G,1}^{n+1} & \dots & a_{G,G}^{n+1} \end{pmatrix}$$

accounts for the coupling between multi-group fluxes. The operator matrices $A_{1,2}^{n+1}(G\times L)$ and $A_{2,1}^{n+1}(L \times G)$ represent the coupling between the fluxes and the precursors' concentrations. The $L \times L$ operator matrix $A_{2,2}$ is diagonal with entries of the form $\frac{\mathrm{Id}}{\delta t} + \lambda_\ell$ that are independent of time. However, $A_{1,1}^{n+1}$, $A_{1,2}^{n+1}$, $A_{2,1}^{n+1}$ are, in fully generality, time-dependent, hence the super-index $n+1$ in the notation. We shall keep the same notation for the matrices before and after the spacial discretization.

Note that problem (5.2) is not fully discretized yet because a spatial discretization needs to be specified. In this respect, the solver that we are employing in our numerical computations is based on a $\mathbb{P}_1$ spatial discretization (on a tetrahedral mesh). Therefore in the fully discretized setting, if $\mathcal{N}_{\boldsymbol{r}}$ are the number of degrees of freedom for the spatial discretization, the dimensions of $\boldsymbol{\phi^{n+1}}$ and $\boldsymbol{C^{n+1}}$ will be $G \times \mathcal{N}_{\boldsymbol{r}}$ and $L\times\mathcal{N}_{\boldsymbol{r}}$ respectively. Thus, the total number of degrees of freedom is $\mathcal{N} = (G+L)\times\mathcal{N}_{\boldsymbol{r}}$. The dimension of, e.g., $A_{1,1}^{n+1}$ will be $G\mathcal{N}_{\boldsymbol{r}} \times G\mathcal{N}_{\boldsymbol{r}}$ and similarly for the rest of the matrices.

In order to estimate the accuracy over $[0, T]$ of the sequential fine solver (5.2) with respect to the exact solution (that we do not know), we choose to determine it by comparing, for $0 \le n \le \underline{\underline{\mathrm{n}}}$, the solution given by $\mathcal{F}_{n\delta t}^0(u_0)$ (where (5.2) is directly inverted) with the solution of an "ultra-fine" solver $\widehat{\mathcal{F}}_{n\delta t}^0$ that uses a time step $\widehat{\delta t} = \delta t/100 = 10^{-4}$ s. (and that also directly inverts (5.2)). We have then estimated $\varepsilon_{\mathcal{F}}$ with

$$\varepsilon_{\mathcal{F}} \approx \max_{0\le n\le \underline{\underline{\mathrm{n}}}} \frac{\|\mathcal{F}_{n\delta t}^0(u_0) - \widehat{\mathcal{F}}_{n\delta t}^0(u_0)\|_{\ell_2(\mathbb{R}^{\mathcal{N}})}}{\|\widehat{\mathcal{F}}_{n\delta t}^0(u_0)\|_{\ell_2(\mathbb{R}^{\mathcal{N}})}} = 3.510^{-2}.$$

This reveals the level of accuracy of the time discretization we want to achieve.

Let us say that now we are interesting in solving, the same problem as (5.1), discretized as in (5.2) but with varying parameters. In this case we assume that instead of a unique $A_{1,1}$, we actually have a family of these, e.g. like $\tilde{A}_{1,1} = (1+\eta)A_{1,1}$ and solving a bunch of them altogether. If $\boldsymbol{\phi^{n,j}(r)}$ and $\boldsymbol{C^{n,j}(r)}$ are the approximations of $\boldsymbol{\phi^n(r)}$ and $\boldsymbol{C^n(r)}$ at the j-th iteration, our iterative scheme reads for $j = 1, \dots, J_{n+1}^*$,

$$\begin{aligned} &\begin{pmatrix} A_{1,1}^{n+1} & 0 \\ 0 & A_{2,2} \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi^{n+1,j}} \\ \boldsymbol{C^{n+1,j}} \end{pmatrix} = \\ &\qquad - \begin{pmatrix} \eta A_{1,1}^{n+1} & A_{1,2}^{n+1} \\ A_{2,1}^{n+1} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi^{n+1,j-1}} \\ \boldsymbol{C^{n+1,j-1}} \end{pmatrix} + \frac{1}{\delta t} \begin{pmatrix} \boldsymbol{\phi^{n,J_n^*}} \\ \boldsymbol{C^{n,J_n^*}} \end{pmatrix}. \end{aligned} \tag{5.3}$$

Note that for different values of η, we can invert the matrix on the left hand side once and get the different values $\left(\boldsymbol{\phi^{n+1,j}}, \boldsymbol{C^{n+1,j}}\right)^T = \left(\boldsymbol{\phi^{n+1,j}}(\eta), \boldsymbol{C^{n+1,j}}(\eta)\right)^T$ associated to different values of η rapidly. As $\eta \in [0, 1]$ grows, the convergence of this internal iterative scheme rapidly increases. Note that for $\eta = 0$, (5.3) is then the solver that was used in [3] that only required 3 to 4 iterations to converge at each time step.

Let us now describe how to solve problem (5.2) with the parareal scheme (2.3) with a reduced number $J < J^*_{n,k}$ of internal iterations in the fine solver. In our case, $U^N_{k,J}$ is a pair consisting of the parareal flux and precursors' concentrations solution at iteration k and at time T_N. With the notations of section 3, $\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J})$ starts at time $T_N = t^N_0$ and reaches $T_{N+1} = t^N_{\underline{\underline{\mathrm{n}}}}$ by performing $\underline{\underline{\mathrm{n}}}$ propagations with a time step of δt. The solution $\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J})$ can therefore be written in our case as

$$\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}(U^N_{k,J}) = \begin{pmatrix} \boldsymbol{\phi}^{\underline{\underline{\mathrm{n}}}}_{\boldsymbol{k,J}} \\ \boldsymbol{C}^{\underline{\underline{\mathrm{n}}}}_{\boldsymbol{k,J}} \end{pmatrix}. \tag{5.4}$$

The intermediate states that are performed to reach $\widetilde{\mathcal{F}}^{T_N}_{J,\Delta T}$ are the following: for $n = 0, \ldots, \underline{\underline{\mathrm{n}}} - 1$ and $j = 1, \ldots, J$

$$\begin{pmatrix} A^{n+1}_{1,1} & 0 \\ 0 & A_{2,2} \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi^{n+1,j}} \\ \boldsymbol{C^{n+1,j}} \end{pmatrix} = - \begin{pmatrix} \eta A^{n+1}_{1,1} & A^{n+1}_{1,2} \\ A^{n+1}_{2,1} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi^{n+1,j-1}} \\ \boldsymbol{C^{n+1,j-1}} \end{pmatrix} + \frac{1}{\delta t} \begin{pmatrix} \boldsymbol{\phi^{n,J}} \\ \boldsymbol{C^{n,J}} \end{pmatrix},$$

with

$$\begin{pmatrix} \boldsymbol{\phi^0_{k,J}} \\ \boldsymbol{C^0_{k,J}} \end{pmatrix} = U^N_{k,J}.$$

The internal iterations are initialized by adapting formulae (3.3) and (3.4) to our case.

5.3. Numerical results. In what follows, we want to illustrate the behavior of our balanced parallel algorithm, this is why, for the sake of convenience, we shall identify one value for η that represents a general feature of iterative algorithms, and we focus on this particular value.

The parameters η and $\varepsilon_{\text{inner}}$ of the internal iterative scheme (5.3) have thus been fixed to preserve this accuracy $\varepsilon_{\mathcal{F}}$ and with a number of iterations that is similar to the one observed in industrial solvers. Since we are using an Euler-backward scheme, the first feature is satisfied by taking $\varepsilon_{\text{inner}} = \varepsilon_{\mathcal{F}}/\underline{\underline{\mathrm{n}}} \approx 10^{-2}/400 \approx 10^{-5}$. For the second, we have benefited from our previous experience with an industrial solver called MINARET (see [9]): we have observed that, by taking $\eta = 0.05$, the number J^*_n of internal iterations is similar to the number of internal iterations involved in our previous works with MINARET : the convergence being achieved whenever the residual between two iterations

$$\frac{\| \left(\boldsymbol{\phi^{n,j}}, \boldsymbol{C^{n,j}}\right)^T - \left(\boldsymbol{\phi^{n,j-1}}, \boldsymbol{C^{n,j-1}}\right)^T \|_{\ell_2(\mathbb{R}^{\mathcal{N}})}}{\| \left(\boldsymbol{\phi^{n,j}}, \boldsymbol{C^{n,j}}\right)^T \|_{\ell_2(\mathbb{R}^{\mathcal{N}})}} \tag{5.5}$$

goes below a prescribed tolerance $\varepsilon_{\text{inner}}$. Note that when $\eta < 0.05$ (resp. $\eta > 0.05$) too few (many) internal iterations are required

In Table 1 and Figure 2 are summarized the values of the parameters employed in these computations.

Since internal iterations are the basic step in all the different methods below, complexity can be evaluated in terms of the total number of internal iterations over $[0, T]$. In the case of the sequential fine solver $\mathcal{F}^0_{J^*,T}(u_0)$, the complexity is

$$\mathcal{C}_{\text{seq}} = \sum_{n=1}^{\underline{\underline{\mathrm{n}}}} J^*_n = 4949.$$

Time parameters	$T = 40$ s., $\Delta T = 5$ s., $\delta t = 0.1$ s. $\underline{N} = T/\Delta T = 8$, $\underline{\mathrm{n}} = \Delta T\ \delta t = 50$, $\underline{\underline{\mathrm{n}}} = T/\delta t = 400$
Sequential fine solver $\mathcal{F}^0_{J^*,T}(u_0)$	Accuracy: $\varepsilon_{\mathcal{F}} = 3.5.10^{-2}$ Internal iterations: $\begin{cases}\varepsilon_{\text{inner}} = 10^{-5}\\ \eta = 0.05\end{cases}$ Behavior of J_n^* (see Figure 2) $\begin{cases}\min_{0\le n\le \underline{\underline{\mathrm{n}}}} J_n^* = 4\\ \max_{0\le n\le \underline{\underline{\mathrm{n}}}} J_n^* = 29\\ \langle J_n^*\rangle = \frac{1}{\underline{\underline{\mathrm{n}}}} \sum_{0\le n\le \underline{\underline{\mathrm{n}}}} J_n^* \approx 12.5\end{cases}$
Truncated fine solver $\mathcal{F}^{T_N}_{J,\Delta T}$	Internal iterations: $\begin{cases}\varepsilon_{\text{inner}} = 10^{-5}\\ \eta = 0.05\end{cases}$ J is a parameter of our study
Coarse solver $\mathcal{G}^{T_N}_{\Delta T}$	Time step of size ΔT No internal iterations: direct inversion of (5.2)
Parareal iterations	Stopping criterion $\varepsilon_{\text{parareal}} \coloneqq \varepsilon_{\mathcal{F}}/10 = 10^{-3}$

Table 1: Summary of the parameters used in the numerical computations.

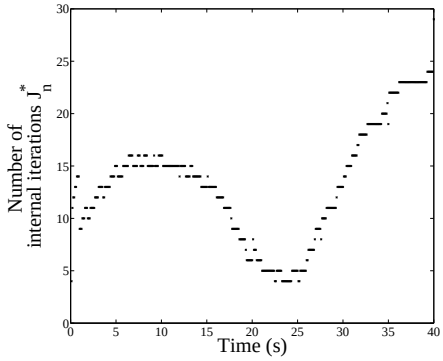

Figure 2: Number of internal iterations J_n^* of the sequential fine solver $\mathcal{F}^0_{J^*,T}(u_0)$.

If we neglect the cost of the coarse propagator, an estimation of the complexity $\mathcal{C}_{\text{plain}}$ and $\mathcal{C}_{\text{trunc}}$ of the plain and truncated parareal schemes is given by the total number of internal iterations over the total number of processors $N_{\text{proc}} = \underline{N} = 8$, namely

$$\mathcal{C}_{\text{plain}} = \frac{\sum_{k=1}^{k_{\text{std}}} \sum_{n=1}^{\underline{\underline{\mathrm{n}}}} J_{n,k}^*}{N_{\text{proc}}} \quad \text{and} \quad \mathcal{C}_{\text{trunc}}(J, \text{Case}) = \frac{\sum_{k=1}^{k_{\text{trunc}}} \sum_{n=1}^{\underline{\underline{\mathrm{n}}}} \min\left(J, J_{n,k}^*\right)}{N_{\text{proc}}}.$$

The value of $\mathcal{C}_{\text{trunc}}$ depends on J and also on the type of initialization of the internal iterations (Case I if we use (3.3) or Case II if we use (3.4)).

In the results presented below, we study:

1. Convergence of the new parareal scheme for different values of J and with the two different types of initializations of the internal iterations (formulas

(3.3)and (3.4)). Convergence will be examined through the study of the errors of the degraded fine propagations with respect to the fine sequential resolution, namely

$$e_k^n = \frac{\| \left(\boldsymbol{\phi_k^{n,J}}, \boldsymbol{C_k^{n,J}}\right)^T - \left(\boldsymbol{\phi^{n,J_n^*}}, \boldsymbol{C^{n,J_n^*}}\right)^T \|_{\ell_2(\mathbb{R}^{\mathcal{N}})}}{\| \left(\boldsymbol{\phi^{n,J_n^*}}, \boldsymbol{C^{n,J_n^*}}\right)^T \|_{\ell_2(\mathbb{R}^{\mathcal{N}})}}, \quad n = 0, \ldots, \underline{\underline{\mathrm{n}}}. \tag{5.6}$$

Convergence will be achieved when

$$\sup_{0 \leq n \leq \underline{\underline{\mathrm{n}}}} e_k^n < \varepsilon_{\text{parareal}},$$

We consider that the choice

$$\varepsilon_{\text{parareal}} \coloneqq \varepsilon_{\mathcal{F}}/10 = 3.5.10^{-3}$$

is tight enough to preserve the accuracy $\varepsilon_{\mathcal{F}}$ of the sequential fine solver in our parareal solutions.

2. An analysis of the acceleration performances will be given through a comparison of the complexity $\mathcal{C}_{\text{trunc}}$ with respect to $\mathcal{C}_{\text{seq}}$. Thus,

$$\mathcal{S}(J, \text{Case}) = \frac{\mathcal{C}_{\text{seq}}}{\mathcal{C}_{\text{trunc}}(J, \text{Case})} \quad ; \quad E(J, \text{Case}) = \frac{\mathcal{S}(J, \text{Case})}{N_{proc}}$$

will be taken as estimations of the speed-up and the efficiency respectively.

3. We will also present some results concerning the use of a reduced basis to limit the storage.

REMARK 5.1. *Note that the errors given in* (5.6) *cannot be used in practice during a calculation as a stopping criterion because the sequential fine resolution will not be carried out. An estimator that could be computed online could involve, e.g., some residual error in the parareal solutions like*

$$r_k = \max_{1 \leq N \leq \underline{N}} \frac{\| \left(\boldsymbol{\phi_k^{N,J}}, \boldsymbol{C_k^{N,J}}\right)^T - \left(\boldsymbol{\phi_{k-1}^{N,J}}, \boldsymbol{C_{k-1}^{N,J}}\right)^T \|_{\ell_2(\mathbb{R}^{\mathcal{N}})}}{\| \left(\boldsymbol{\phi_k^{N,J}}, \boldsymbol{C_k^{N,J}}\right)^T \|_{\ell_2(\mathbb{R}^{\mathcal{N}})}}, \quad k \geq 2. \tag{5.7}$$

However, in this study, we are placing ourselves in an a posteriori validation to carry out the convergence study as accurately as possible and we will therefore analyze the convergence in the parareal iterations through the errors e_k^n.

We start by plotting in Figure 3 the convergence of the plain parareal algorithm. Since we have fixed the convergence criterion of the parareal iterations to $\varepsilon_{\text{parareal}} = 3.5.10^{-3}$, convergence is achieved in $k_{\text{plain}} = 2$ iterations. We have in this case

$$\mathcal{C}_{\text{plain}} = 16050/8 = 2006.2.$$

In other words, with $\underline{N} = 8$ processors, the plain parareal algorithm provides an acceleration of a factor of

$$\mathcal{S}_{\text{plain}} = \mathcal{C}_{\text{seq}}/\mathcal{C}_{\text{plain}} \approx 2,$$

hence only twice faster when using 8 processors ($E = 1/4$). Let us now analyze the convergence of the parareal scheme with truncated internal iterations. The first

type of initialization of the internal iterations (see formula (3.3)) presents a very slow convergence rate. As an example, convergence is not even achieved after $k_{\text{trunc}} = 6$ parareal iterations with $J = 10$ and we are slower than in the plain parareal scheme since we have

$$\mathcal{C}_{\text{trunc}}(10, 1) > 20720/8 = 2590 > \mathcal{C}_{\text{plain}}.$$

This behavior can be well understood by analyzing the condition $\rho^J \leq C\delta t\varepsilon^2$ of theorem 3.2. Indeed, we have evaluated that $\rho \approx 0.94$ during the whole transient and, since we use an implicit Euler scheme, we have $\varepsilon_{\delta\mathcal{G}} \approx \Delta t$. As a result, the condition $\rho^J \leq C\delta t\varepsilon^2_{\delta\mathcal{G}}$ is fulfilled in our case for large values of J. Although one could think that the constant C might be playing an important role here, from the construction of the proof of theorem 3.2 it follows that C is built from order one terms.

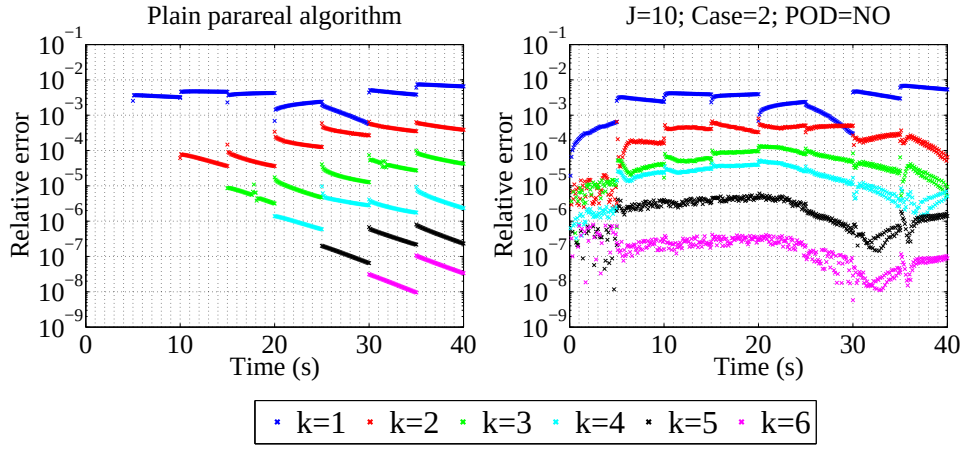

Figure 3: Left: Plain parareal algorithm. Right: Convergence with the second type of initialization ($J = 10$). The legend will also apply for the rest of the plots.

Let us now turn to the second type of initialization proposed in (3.4). The convergence properties of this scheme are significantly improved with respect to the first type of initialization. Indeed, the case $J = 10$ is now performing similarly to the plain parareal algorithm (see Figure 3, right). What is more, by performing only one internal iteration ($J = 1$), parareal convergence in the whole time interval is achieved after $k_{trunc} = 4$ parareal iterations (see Figure 4, left). This means that we only need $J_{\text{cumul}} = J \times k_{trunc} = 1 \times 4 = 4$ internal iterations (one per parareal iteration) to reach a converged state at every fine time t^n, $0 \leq n \leq \underline{\underline{\text{n}}}$. If we compare this with the values of J_n^* displayed in Figure 2, it seems clear that $J_{\text{cumul}} = 4$ is very small in comparison with the values of J_n^*. This illustrates to what extend it is interesting to not only resume the internal iterations from $k - 1$ as the first initialization does, but also to take the dynamics of the process into account as this second initialization is doing. We have in this case

$$\begin{cases} \mathcal{C}_{\text{trunc}}(1, 2) = 1 \times 6 \times 400/8 = 200 \\ E_{\text{trunc}}(1, 2) = \dfrac{4949}{200 \times 8} \approx 3.1 \end{cases}$$

This value of the efficiency $E_{\text{trunc}}(1,2)$ implies that, if we implemented this parareal scheme in only one processor, the resolution would be 3.1 times faster (and with the same accuracy $\varepsilon_{\mathcal{F}}$) than the traditional sequential procedure with the fine solver $\mathcal{F}^0_{J^*,T}$. We recover here one of the basic features of spatial domain decomposition in which computations on a single processor can be accelerated with these algorithms.

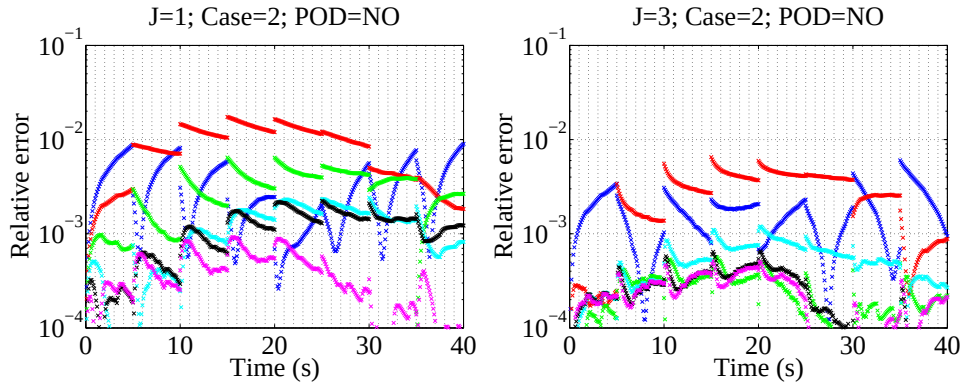

Figure 4: Convergence with the second type of initialization.

Despite the high efficiency result, the value $k_{\text{trunc}} = 4$ represents a relatively large degradation with respect to the number $k_{\text{plain}} = 2$ iterations of the plain parareal scheme. We can "soften" the degradation by increasing the total number of internal iterations. In the case $J = 3$, convergence is achieved in $k_{\text{trunc}} = 3$ iterations and we still have competitive efficiency results:

$$\begin{cases} \mathcal{C}_{\text{trunc}}(3,2) = 3051/8 \approx 381.4 \\ E_{\text{trunc}}(3,2) = \dfrac{4949}{381.4 \times 8} = 1.6 \end{cases}$$

We believe that these efficiency results are very encouraging, since they represent a dramatic improvement of the performances of the plain parareal algorithm and makes the parallelization of time be a competitive option to for the exploitation of massively parallel computers. However, we would like to point out at this stage that the study of the actual performances of the method should be carried out with computations of the residual (5.7) rather than with the errors (5.6) as in the present discussion. In this case, we usually expect one additional parareal iteration to detect convergence with residuals like (5.7). As a result, we expect the efficiency to be degraded with respect to the present results, but it seems that there is still room to keep it larger than one (or, at least, very close to it).

It is now important to analyze whether the convergence properties of the scheme are degraded if the initial values of the internal iterations are replaced by surrogates coming from a reduced basis X_M of small dimension M (see (4.1)). We present results in the case $J = 3$ and with the second type of initialization (similar conclusions could be drawn from other configurations). In the left plot of Figure 5 are presented calculations with a reduced basis of dimension $M = 2$ that has been built with a size of the moving window of $M_{\max} = 8$. The accuracy of the compression is $\varepsilon_{\text{compr}} \approx 5.10^{-4}$. One can observe that convergence is still achieved in $k_{\text{trunc}} = 3$ parareal iterations,

which proves that the properties of the scheme are not degraded very much if the compression is "tuned" in a proper way.

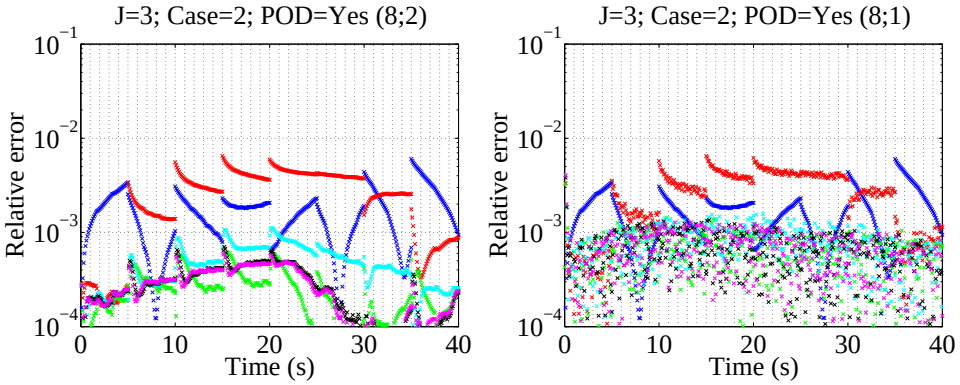

Figure 5: Convergence in the second type of initialization with $J = 3$ and compression of the starting guesses. On the left, $M = 2$, $M_{\max} = 8$ and $\varepsilon_{\text{compr}} < \varepsilon_{\text{parareal}}$. On the right, $M = 1$, $M_{\max} = 8$ and $\varepsilon_{\text{compr}} > \varepsilon_{\text{parareal}}$.

From what has been observed in our computations, the key of this "tuning" relies in the relation between the compression $\varepsilon_{\text{compr}}$ and the accuracy of the fine solver $\varepsilon_{\mathcal{F}}$. The compression should preserve the accuracy of the fine solver over time intervals of size ΔT. Since, at each fine time step we add an error of order $\varepsilon_{\text{compr}}$, it means that we should have

$$\varepsilon_{\text{compr}} < \frac{\varepsilon_{\mathcal{F}}}{\Delta T/\delta t} = \frac{\varepsilon_{\mathcal{F}}}{\underline{\mathrm{n}}} = 7.10^{-4},$$

which is a condition that is satisfied in the previous example. Let us now consider an example in which $\varepsilon_{\text{compr}} \geq \varepsilon_{\mathcal{F}}/\underline{\mathrm{n}}$. For that, we consider a moving-window POD with $M = 1$ and $M_{\max} = 8$, whose compression accuracy is $\varepsilon_{\text{compr}} = 10^{-2}$. As the right plot of Figure 5 shows, convergence properties are severely degraded. Indeed, although most of the errors are below $\varepsilon_{\text{parareal}} = 3.5.10^{-3}$ after 3 parareal iterations, convergence stagnates at a level of around 2.10^{-3}, which is too close to the limit $\varepsilon_{\text{parareal}}$ to confirm convergence without any reservations.

6. Conclusions and future work. We have introduced the main foundations of the SBPM : a version of parareal scheme that involves a degraded fine solver with non converged internal iterations at every time step combined with a reduced storage of the time history of the incremental approximated solution. We have provided theoretical evidence of the convergence of the scheme and have studied its behavior in a numerical application related to the field of neutronics. The numerical results show a significant improvement of the parallel performances with respect to the plain parareal algorithm and we believe that it is an important step towards making the parallelization of time be a fully competitive option for the exploitation of massively parallel computers. In particular, our example show that, even implemented in on a serial platform, the SBPM is faster than the plain implementation of the fine solver. The results of this article will be complemented in a future work by an implementation of the method in a solver of the nuclear industry like MINARET. This will prove feasibility in an

involved non-academic setting and a study of the efficiency of the method through residuals of the type of (5.7) could exactly quantify the performances of the method.

REFERENCES

[1] E. Aubanel, *Scheduling of tasks in the parareal algorithm*, Parallel Computing, 37 (2011), pp. 172–182.

[2] Bal, G., *Parallelization in time of (stochastic) ordinary differential equations*, 2003. Preprint, http://www.columbia.edu/ gb2030/PAPERS/paralleltime.pdf.

[3] A.-M. Baudron, J.-J. Lautard, Y. Maday, M. K. Riahi, and J. Salomon, *Parareal in time 3D numerical solver for the LWR Benchmark neutron diffusion transient model*, Journal of Computational Physics, 279 (2014), pp. 67 – 79.

[4] R. Dautray and J.-L. Lions, *Analyse mathématique et calcul numérique*, Masson, 1984.

[5] Martin J Gander, *50 years of time parallel time integration*, in Householder Symposium XIX June 8-13, Spa Belgium, 2015, p. 81.

[6] R. Guetat, *Méthode de parallélisation en temps: Application aux méthodes de décomposition de domaine*, PhD thesis, Paris VI, 2012.

[7] A. Hebert, *Applied reactor physics*, Presses inter Polytechnique, 2009.

[8] S. Langenbuch, W. Maurer, and W. Werner, *Coarse-mesh flux expansion method for the analysis of space-time effects in large light water reactor cores*, Nucl. Sci. Eng., 63 (1977), pp. 437–456.

[9] J.-J. Lautard, Y. Maday, and O. Mula, *MINARET: Towards a parallel 3D time-dependent neutron transport solver*, Submitted.

[10] J.L. Lions, Y. Maday, and G. Turinici, *Résolution d'EDP par un schéma en temps pararéel*, C. R. Acad. Sci. Paris, (2001). t. 332, Série I, p. 661-668.

[11] Y. Maday, *The 'Parareal in Time' Algorithm*, in Substructuring Techniques and Domain Decomposition Methods, F. Magoulès, ed., Saxe-Coburg Publications, 2010, ch. 2, pp. 19–44.

[12] Y. Maday and G. Turinici, *The Parareal in Time Iterative Solver: a Further Direction to Parallel Implementation*, in Domain Decomposition Methods in Science and Engineering, Springer Berlin Heidelberg, 2005, pp. 441–448.

[13] M. Minion, *A hybrid parareal spectral deferred corrections method*, Comm. App. Math. and Comp. Sci., 5 (2010).

[14] O. Mula, *Some contributions towards the parallel simulation of time dependent neutron transport and the integration of observed data in real time*, PhD thesis, Paris VI, 2014.

[15] A. Toselli and O. Widlund, *Domain decomposition methods: algorithms and theory*, vol. 3, Springer, 2005.